Contribuții la interacțiunea multimodală în 3D

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea Politehnica Timişoara
în domeniul Calculatoare și Tehnologia Informației
de către

Ing. Stelian-Nicolae NICOLA

Conducător ştiinţific: prof.univ.dr.ing. Lăcrămioara STOICU-TIVADAR
Referenţi ştiinţifici: prof.univ.habil.dr.ing. Mihnea Alexandru MOISESCU
prof.univ.dr. inf. Viorel NEGRU
prof.univ.dr.ing. Radu VASIU

Ziua susţinerii tezei: 31 iulie 2023

Cuvânt înainte

Teza de doctorat a fost elaborată pe parcursul activității mele în cadrul Departamentului de Automatică și Informatică Aplicată al Universității Politehnica Timișoara.

Mulțumiri deosebite se cuvin conducătorului de doctorat prof. univ. dr. ing. Lăcrămioara STOICU-TIVADAR care a crezut în mine și mi-a acordat sprijin în elaborarea și susținerea acestei teme de cercetare.

De asemenea adresez mulțumiri domnilor profesori din cadrul comisiei de îndrumare din cadrul programului doctoral: prof. univ. dr. fiz. Gheorghe MIHALAȘ, prof. univ. dr. ing. Ioan Silea și prof. univ. dr. ing. Vasile STOICU-TIVADAR, și ale celor doi specialiști din domeniu: Conf. dr. ing. Ciprian-Bogdan CHIRILA și Ș.l. dr. ing. Dorin BERIAN, ale căror sfaturi teoretice și practice mi-au fost de un real folos pe parcursul programului de doctorat.

Doresc sa mulțumesc colectivului din care fac parte, care a fost alături de mine și m-a sprijinit în elaborarea anumitor soluții din cadrul actualei teze.

Mulțumesc familiei mele care a fost alături de mine și mi-a sprijinit acest drum, ajutându-mă cu orice a fost nevoie.

Timişoara, iunie 2023 Stelian-Nicolae NICOLA

CUPRINS

Notaţii, abrevieri, acronime

VR	Realitate Virtuală
AR	Realitate Augumentată
MR	Realitate Mixtă
XR	Realitate Extinsă/Experimentală
LM	Leap Motion
IT	Tehnologia Informației
PC	Personal Computer
USB	Universal Serial Bus
KNN	K-nearest neighbors
SVM	Support Vector Machine
DNN	Deep Neural Network
CNN	Convolutional Neural Network
ASL	American Sign Language
MLP	Multi Layer Perceptron
RNN	Recurrent Neural Network
SHREC	Shape Retrieval Contes
ANN	Artificial Neural Network
Bi-GRU	Bidirectional Recurrent Unit
Bi-LSTM	Bidirectional Long Short Term Memory
LDA	Linear Discriminant Analysis
RF	Random Forest
NB	Naive Bayes
AB	Ada Boost
SFM	Social Force Model
BPNN	Back Propagation in Neural Network
LSTM	Long Short Term Memory
TCM	Temporal Convolutional Network
HMM	Hidden Markov Model
DT	Decision Tree
ML	Machine Learning
RBF SVM	Radial Basis Function Support Vector Machine
SVC	Support Vector Classification
LR	Logistic Regression
CART	Decision Tree Classifier
AP	Adevărat Pozitiv
AN	Adevărat Negativ
FP	Fals Pozitiv
FN	Fals Negativ
P	Precizia
R	Funcția de Rechemare
A	Acuratețea
F1	Scorul f1
ADN	Acidul Dezoxiribonucleic

Lista de tabele

Lista de figuri

1. INTRODUCERE

1.1. Introducere

Teza de doctorat „Contribuții la interacțiunea multimodală în 3D” are ca obiectiv principal **interacțiunea în spațiul virtual 3D**, expus prin diferite metode utilizatorilor. Principala interacțiune cu spațiul 3D se referă la **interacțiunea prin gesturi**. Scopul cercetării îl reprezintă crearea de noi gesturi utilizate în diferite domenii pentru a veni în ajutorul educației (creșterea interactivității dintre om și calculator), medicină (recuperarea mâinii și a brațului), ingineriei (controlul unor roboți) și informatică (învățarea unor algoritmi de sortare). Modul în care spațiul 3D este afișat și/sau explorat reprezintă o altă parte a tezei de doctorat. Pentru o vizualizare/interacțiune cât mai bună cu spațiul virtual 3D se utilizează tehnologii bazate pe realitate virtuală (VR), augumentată (AR) sau mixtă (MR).

Teza de doctorat se concentrează pe crearea și interpretarea gesturilor făcute cu mâna pentru a fi detectate de un dispozitiv de intrare numit Leap Motion (LM). Gesturile sunt considerate statice sau dinamice urmarea aplicării unor tehnici de clasificare. Algoritmii dezvoltați pentru recunoașterea anumitor gesturi au la bază formule matematice de calcul ale distanțelor, unghiurilor, direcțiilor și suprafețelor în spațiul virtual 3D. O altă caracteristică importantă legată de crearea de gesturi o reprezintă precizia și calitatea gestului recunoscut. Astfel, pentru a avea o precizie cât mai bună în clasificarea gesturilor create s-au folosit rețele neuronale convoluționale. Acuratețea în detecție/clasificare a gestului a crescut, iar unele inconveniențe (detectarea greșită a mâinii stângi sau drepte) au fost eliminate.

Gesturile create sunt aplicate în domeniul educațional prin aplicații 3D create să vină în ajutorul utilizatorilor. Interactivitatea pe astfel de aplicații care folosesc alte moduri de folosire ale acestora față de cele clasice (mouse, tastatură, atingere – pentru telefoane mobile) crește. Au fost create diferite aplicații 3D educaționale în care utilizatorul poate să învețe oasele scheletului uman, aminoacizii și nucleotidele din ADN, folosind gesturi. Folosirea gesturilor în manipularea aplicațiilor precum și utilizarea conceptelor de realitate virtuală și/ sau augumentată face ca utilizatorul să fie mai aproape de aplicație, iar concentrarea acestuia crește prin trăirea senzației de prezență fizică în interiorul aplicației.

Un alt domeniu în care au fost aplicate gesturile noi create îl reprezintă domeniul medical, cu aplicare în recuperare. Au fost creați algoritmi pentru detecția gesturilor folosite în recuperare mâinii și a brațului. Modul în care aplicațiile 3D create asistă pacienții în recuperare poate să le includă în noua categorie apărută de aparate în recuperare, așa numitele aparate bazate pe antrenare sau dispozitive bazate pe antrenare. Acestea având un rol fundamental în asistarea virtuală a pacientului. Monitorizând și asistând pacienții prin astfel de aparate/aplicații, gesturile create au fost împărțite în sub-gesturi, deoarece este esențială o vizualizare a progresului de recuperare a mâinii în timp.

Calitatea gesturilor create pentru a fi folosite în aceste domenii este foarte importantă. Astfel pentru mai bună detectare și clasificare a gesturilor s-au folosit

algoritmi matematici de calcul al distanțelor, direcțiilor, ariilor, unghiilor pentru diferite puncte ale mâinii. În plus s-au utilizat rețele neuronale pentru clasificarea gesturilor dinamice implicate în aplicații.

O altă arie abordată în această teză este folosirea realității virtuale (VR) și/sau augumentate (AR) și/sau mixte (MR) pentru o mai bună interacțiune dintre om și mașină. Astfel au fost dezvoltate aplicații care se bazează pe aceste tehnologii și s-a studiat modul de interacțiune pe acestea. Diferite metrici au fost dezvoltate pentru măsurarea interactivității în cadrul aplicațiilor 3D. Metricile definite sunt prezente și în alte aplicații educaționale, dar ca tip de serviciu utilizat, s-a automatizat detecția interactivității pe aplicații 3D bazate pe VR/AR/MR. Punctele personale adăugate la măsurarea interactivității pe aplicații sunt metrici adaptate pe astfel de aplicații care implică o utilizare constantă pe o durată de timp limitată. Principalele metrici folosite în măsurarea interactivității au fost: tip selecție (buton, obiect 3D), timp și viteză. Aceste metrici sunt prezente și într-un plan de testare clasic, dar în abordarea propusă de mine acestea sunt integrate în aplicații. Se elimină partea de subiectivism în ceea ce privește interactivitatea prin folosirea acestui nou tip de măsurare și detecție a interacțiunii dintre om și mașină/dispozitiv.

Ca o concluzie a celor prezentate mai sus, în teza de doctorat sunt propuse mai multe abordări pentru interacțiunea dintre om și calculator: interacțiunea prin gesturi cu concentrare pe calitatea gesturilor create, modul de prezentare/vizualizare a aplicațiilor 3D, dar și măsurarea interactivității pe astfel de aplicații.

1.2. Obiectivele tezei

Principalul scop al acestei teze se concentrează pe gesturi create și folosite în diverse aplicații pentru diferite domenii: educațional, medical, ingineresc. Un al scop este și folosirea noilor tehnologii VR/AR/MR pentru o mai bună interacțiune dintre om și mașină. Toate acestea aduc contribuții importante la domeniul Calculatoare și Tehnologia informației.

În continuare vor fi prezentate principalele beneficii și contribuții aduse prin dezvoltarea acestor gesturi și folosirea acestor tehnologii pe diferite aplicații construite:

Beneficii în domeniul Calculatoare și Tehnologia Informației:

- Compararea în literatura de specialitate a modelelor de clasificare și a modurilor de clasificare a gesturilor
- Identificarea tehnologiilor care pot fi utilizate în creșterea interactivității om-calculator
- Crearea a 7 algoritmi pentru identificarea gesturilor dinamice LM
- Definirea parametrilor ce descriu gesturile
- Crearea unui set de date pentru identificarea a 3 gesturi dinamice
- Clasificarea gesturilor dinamice prin diferite modele/clasificatoare de rețele neuronale

Beneficii în domeniul medical prin folosirea gesturilor și a tehnologiilor:

- Creșterea interactivității om-calculator
- Crearea de gesturi normale/clasice utilizate în diferite acțiuni pentru manipularea obiectelor virtuale 3D
- Creșterea calității gesturilor definite în acest scop folosind algoritmi de clasificare
- Definirea de gesturi statice și dinamice pentru a fi recunoscute de dispozitivul LM
- Creșterea acurateței de detecție a gestului pentru dispozitivul LM
- Vizualizare mai bună a datelor prin dispunerea acestora sub formă 3D și prin utilizarea VR

Beneficii în domeniul educațional:

- Aplicație interactivă pentru învățarea algoritmilor de sortare, învățarea aminoacizilor sau învățarea diferitor sisteme anatomice umane
- Folosirea de concepte noi utilizate în aplicații educaționale – gamificarea
- Interfețe apropiate de utilizator prin folosirea tehnologiilor bazate pe VR/AR/MR
- Gesturi definite în scopul creșterii interactivității în astfel de aplicații

Beneficii în domeniul medicinii recuperatorii

- Gesturi create în scopul recuperării mâinilor
- Sub-gesturi create pentru o evoluție cât mai bună a recuperării, recuperare graduală
- Clasificarea riguroasă a sub-gesturilor create folosind algoritmi de rețele neuronale
- Aplicații care asistă utilizatorul în recuperare, astfel recuperarea se poate face oriunde și oricând
- Creșterea gradului de dificultate ale gesturilor în funcție de afecțiunea utilizatorilor

Beneficii privitoare la evaluarea aplicațiilor 3D

- Crearea de noi metrici folosite pentru măsurarea în mod dinamic a interacțiunii aplicaților 3D
- Măsurarea în mod dinamic a interacțiunii pe o aplicație 3D
- Eliminarea subiectivismului prin testarea automată a folosinței unor funcționalități oferite în aplicații
- Monitorizarea funcționalităților oferite în aplicații 3D și creșterea utilizării acestora

1.3. Structura tezei

Lucrarea de doctorat "Contribuții la interacțiunea multimodală în 3D" este împărțită în 7 capitole. În cele ce urmează vor fi prezentate capitolele tezei și vor fi descrise principalele idei care se desprind din fiecare capitol în parte.

Capitolul 1 cuprinde o introducere despre cele realizate în teza de doctorat fiind prezentată tema acesteia, "*Contribuții la interacțiunea multimodală în 3D*". S-au identificat care sunt principalele contribuții aduse în domeniul Calculatoare și Tehnologia Informației prin construirea de noi gesturi care să poată fi utilizate pe interfețe 3D. Sunt prezentate principalele domenii în care sunt folosite gesturile împreună cu beneficiile utilizării acestora. Totodată se face referire la tehnologiile utilizate VR, AR și MR pe astfel de interfețe.

Capitolul 2. Acest capitol este destinat prezentării principalelor rezultate care au fost găsite în literatura de specialitate. Astfel acesta este structurat în 2 mari subcapitole în care se prezintă starea actuală în recunoașterea și prelucrarea gesturilor și starea actuală a aplicațiilor 3D bazate pe VR, AR, MR. La sfârșitul fiecărui subcapitol este făcută o analiză critică asupra lucrărilor/proiectelor/aplicațiilor studiate.

Capitolul 3. Principalele tehnologii software utilizate în dezvoltarea aplicațiilor din teza de doctorat sunt prezentate în capitolul 3. Tot aici se face o categorisire a metodelor de prelucrare a gesturilor folosind formule matematice sau rețele neuronale. Sunt prezentate cum sunt construite interfețele 3D pentru a putea fi utilizate gesturile dezvoltate, precum și tehnologiile utilizate pentru dezvoltarea de aplicații bazate pe VR, AR și MR.

Capitolul 4. Contribuțiile principale în aplicațiile 3D bazate pe gesturi Leap Motion sunt prezentate în acest capitol. Algoritmii dezvoltați în recunoașterea gesturilor statice și dinamice sunt expuși prin prezentarea unor scheme bloc pe care se bazează aceștia. Algoritmii dezvoltați sunt însoțiți de principalele formule matematice utilizate în detecția și procesarea gesturilor.

Capitolul 5. Acest capitol face referire la algoritmii de clasificare a gesturilor folosind rețele neuronale. Descrierea gesturilor prin diferite caracteristici și colectarea acestora într-un set de date este abordată la început. Apoi se trece la filtrarea datelor și clasificarea gesturilor descrise. La sfârșitul acestui capitol se face o comparație a modelelor utilizate în clasificarea gesturilor, oferindu-se care sunt principalele modele de rețele neuronale care pot fi utilizate pentru clasificarea gesturilor definite pe un set de date generat.

Capitolul 6. Modul cum sunt aplicați algoritmii de detecție/prelucrare a gesturilor precum și aplicarea interfețelor bazate pe VR, AR și MR prezentate în capitolele anterioare, sunt explicați în dezvoltarea și folosirea aplicațiilor educaționale și aplicațiilor de recuperare. Principalele aplicații dezvoltate pe aceste două domenii: educațional și medicină recuperatorie sunt prezentate în acest capitol. La sfârșitul acestuia se face o comparație a modului de captare a interacțiunii pe aceste tipuri de aplicații.

Capitolul 7. În acest capitol sunt prezentate concluziile tezei de doctorat, axate pe ceea ce s-a realizat atât pe ramura gesturilor cât și pe interacțiunea pe aplicații 3D. Principalele direcții de continuare a cercetării în acest domeniu, precum

și diferitele domenii în care se mai pot face îmbunătățiri sunt prezentate la sfârșitul acestui capitol.

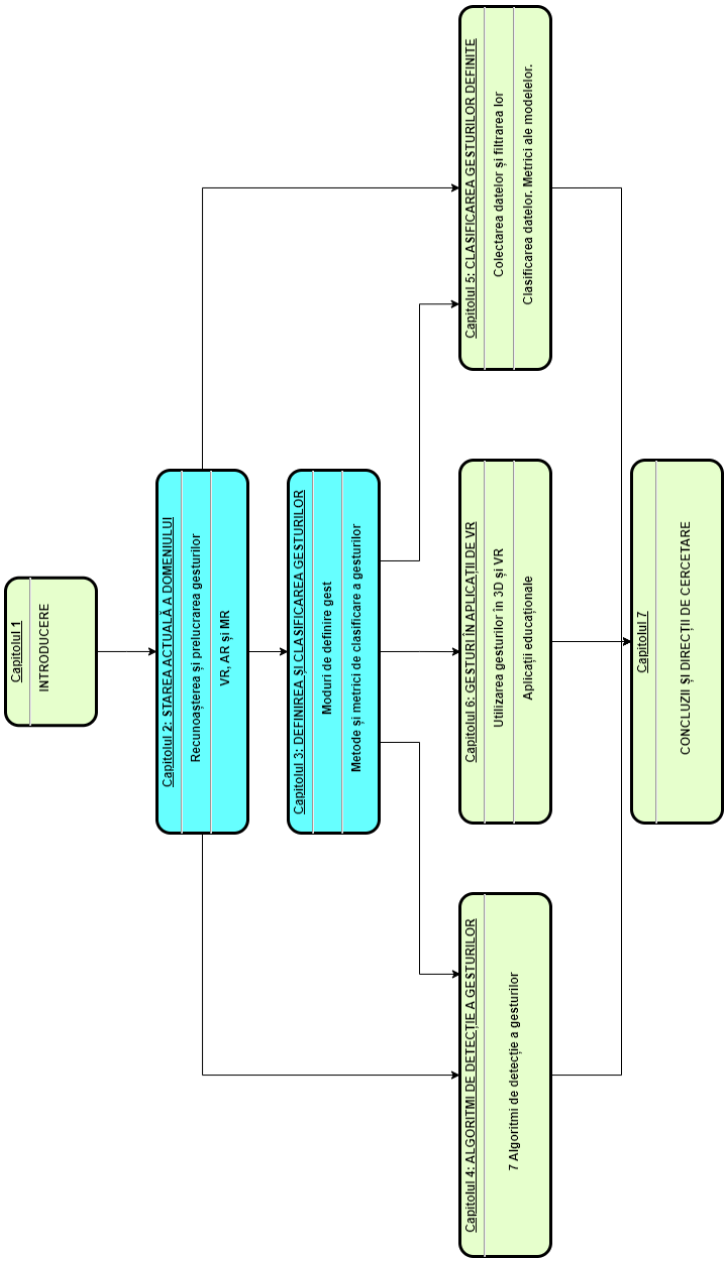

Figura 1.1. Structura tezei

2. STAREA ACTUALĂ A DOMENIULUI

În acest capitol voi prezenta care sunt preocupările din literatura de specialitate în ceea ce privește tema de doctorat. Astfel acest capitol va fii împărțit în 3 părți.

- Starea actuală în recunoașterea și prelucrarea gesturilor.
- Starea actuală în domeniul realității virtuale (VR), augumentate (AR) și mixte (MX)
- Concluzii și perspective după studierea cercetărilor din domeniu.

2.1. Starea actuală în recunoașterea și prelucrarea gesturilor

Conform dicționarului *„Merriam Webster"* gestul este definit ca o mișcare de obicei a corpului sau a membrelor care poate să exprime sau să accentueze o idee, un sentiment/stare, o acțiune sau o atitudine [MW23].

Dex-ul limbi române definește gestul ca o mișcare a mâinii, a capului etc. ce exprimă o idee, un sentiment , o intenție care înlocuiește în unele cazuri vorbele, dând mai multă expresivitate vorbirii [DEX23].

Printre gesturile făcute cu membrele superioare (cu mâinile) acestea se pot împărții în două categorii: gesturi statice și gesturi dinamice. Gesturile dinamice reprezintă mișcarea mâinilor sau a mâinii într-un interval de timp stabilit și pe o anumită distanță. Printre cele mai cunoscute gesturi dinamice sunt gesturile alfanumerice (gesturi ce exprimă numere sau litere). Cele statice exprimă o stare sau chiar un număr, acestea nu pot fi definite pe o durată de timp. Ca și exemplu pentru acest tip de gesturi se pot amintii gestul de aprobare sau dezaprobare și gest prin care se arată un număr prin ridicarea unui număr de degete.

În zilele noastre domeniile care folosesc gesturile în desfășurarea anumitor proceduri sau acțiuni sunt: domeniul medical (proceduri chirurgicale, recuperare, antrenare), ingineresc (IT, mecanică, robotică – controlul uni braț robotic [LI19]).

S-au făcut anumite cercetării în domeniul medical (medici chirurgi) pentru a se observa care sunt principalele acțiuni desfășurate de aceștia în timpul unei proceduri. Este important ca în acest domeniu să fie implementate gesturi în desfășurarea unor proceduri pentru a scutii munca chirurgilor. Anumite acțiuni care pot fi înlocuite cu gesturi se pretează a fi abordate. Ca exemplu de interacțiune dintre chirurg și calculator este selectarea unei zone dintr-o imagine. În mod normal chirurgul trebuie să o desfășoare prin selectarea și apăsarea cu mouse-ul unui calculator a unei zone din imagine. Aceasta presupune ca acesta să se oprească din acțiunea desfășurată pe pacient și să folosească calculatorul. Atenția acestuia fiind concentrată la calculator nu la pacient sau la o anumită zonă în care se desfășoară procedura. Folosirea gesturilor sau a comenzilor vocale în detrimentul mouse-ului sau a tastaturii (echipamente de intrare periferice de bază) pentru anumite acțiuni este recomandată.

Mai multe gesturi desfășurate de către chirurgi [SA-N19], [CHO18] la anumite proceduri sunt detectate în mai multe lucrări științifice. O primă lucrare care tratează

numeroase acțiuni desfășurate prin gesturi la o operație chirurgicală este tratată de cercetătorii tunisieni [AME16], [1AME20]. Principalele acțiuni detectate de aceștia au fost: *click* – pentru selecție, *rotire* – pentru rotirea imaginilor în sensul acelor de ceasornic sau invers, *schimbarea contrastului* – pentru creșterea sau scăderea contrastului imaginilor, *mărire/micșorare* – pentru mărirea sau micșorarea imaginilor, *mișcare* – pentru mișcarea imaginilor la stânga sau la dreapta respectiv *înainte* și *înapoi* – pentru căutarea imaginilor din secvențe.

Pentru aceste acțiuni descrise mai sus au fost dezvoltate 11 gesturi făcute cu mâna. Dispozitivul sau senzorul folosit pentru detectarea acestor gesturi este Leap Motion [LM16]. Senzorul LM este un dispozitiv echipat cu 2 camere monocromatice și 3 led-uri cu infra-roșu pentru vederea tri-diemnsională. Nmărul de cadre pe care camerele acestui dispozitiv le poate realiza este de maxim 300 de cadre pe secundă în funcție de performanțele computer-lui la care acesta este conectat. La o setare inițială pentru un computer obișnuit acest număr de cadre este egal cu 120. Senzorul LM este prevăzut cu o ieșire USB pentru conectarea la PC/laptop. Software-ul acestui dispozitiv poate recunoaște fiecare deget și joncțiune de degete de la mâini. În plus acesta are predefinit 4 gesturi: gest care simulează apăsarea unui ecran (Screen_Tap), gestul de glisare a mâinii (Swipe), gest ce simulează apăsarea unei taste (Key_Tap) și gestul ce semnifică rotirea unui buton (Circle). Se pot definii noi gesturi care să fie recunoscute de acest dispozitiv. Astfel se pot crea baze de date care să permită o descriere riguroasă a unui nou gest definit. De exemplu în aceste baze de date se pot stoca: poziții, distanțe sau unghiuri în spațiul 3D.

În tabelul 2.1.1 vor fi explicate gesturile dezvoltate de către acest grup de cercetători.

Nume gest	**Descriere gest (degete implicate și mod de folosire)**
Click (gest de selectare)	- pentru acest gest se folosește degetul arătător și mare, iar celelalte degete sunt închise în palmă. Gestul este împărțit în 3 faze: în prima fază degetul mare și cel arătător sunt întinse și deschise față de palmă, a doua fază presupune apropierea degetului mare către cel arătător, iar ultima fază este când vârful degetului mare atinge degetul arătător. În toate cele 3 faze degetele neimplicate la acest gest sunt închise în palmă. Prin acest gest se poate selecta o regiune particulara din imagine.
Rotirea la stânga (gest de rotire)	- se folosește doar degetul arătător, celelalte degete fiind strânse în palmă. Pentru a face acest gest utilizatorul trebuie să miște degetul arătător astfel încât să deseneze virtual un cerc. Mișcarea degetului se realizează in sens contrar acelor de ceasornic. Prin acest gest este realizată rotirea imaginii la stânga.
Rotirea la dreapta (gest de rotire)	- pentru acest gest se folosește doar degetul arătător, celelalte degete fiind strânse în palmă. Pentru a face acest gest utilizatorul trebuie să miște degetul arătător astfel încât să deseneze

	virtual un cerc. Mișcarea degetului se realizează în sensul acelor de ceasornic. Prin acest gest este realizată rotirea imaginii la dreapta.
Mărirea contrastului (gest de creștere)	- se folosesc toate degetele de la mână, acestea trebuie să fie deschise. Gestul presupune plecarea de la o poziție 0 care este definită cu poziționarea mâinii pe orizontală și mișcarea/glisarea acesteia în sus până la poziția finală care presupune formarea unui unghi de maxim 90 de grade între poziția inițială și finală. Acest gest este folosit pentru creșterea contrastului imaginii.
Micșorarea contrastului (gest de micșorare/diminuare)	- pentru acest gest se folosesc toate degetele de la mână, acestea trebuie să fie deschise. Gestul presupune plecarea de la o poziție 0 care este definită cu poziționarea mâinii pe orizontală și mișcarea/glisarea acesteia în jos până la poziția finală care presupune formarea unui unghi de maxim -90 de grade între poziția inițială și finală. Acest gest este folosit pentru scăderea contrastului imaginii
Observație: Aceste două gesturi se pot definii ca flexia (mărirea contrastului) si extensia (micșorarea contrastului) mâinii, cu orientare a palmei paralel față de LM.	
Mărire (gest de mărire imagine)	- se folosește toată mâna/toate degetele de la mână. Acesta presupune 3 faze. Faza de start este definită de mâna deschisă/degete întinse fiind folosită pentru alegerea zonei care se dorește a fi mărită, cea dea doua fază presupune închiderea mâinii/strângerea degetelor în palmă, iar când s-a ajuns la mana închisă este atinsă ultima fază. Acest gest este folosit pentru mărirea unei zone din imagine.
Micșorare (gest de micșorare imagine)	- pentru acest gest se folosește toată mâna/toate degetele de la mână. Acesta presupune 3 faze. Faza de start este definită de mâna închisă/degetele strânse în palmă fiind folosită pentru alegerea zonei care se dorește a fi micșorată, cea dea doua fază presupune deschiderea mâinii/întingerea degetelor, iar când s-a ajuns la mana complet deschisă este atinsă ultima fază. Acest gest este folosit pentru micșorarea unei zone din imagine.
Observație: Aceste două gesturi se pot definii ca flexia (mărirea) și extensia (micșorarea) degetelor.	
Mișcare la stânga (gest de mișcare imagine)	- pentru efectuarea acestui gest se folosește degetul arătător deschis, iar toate celelalte degete sunt închise/strânse în palmă. Mișcarea degetului la stânga astfel încât acesta să deseneze virtual o linie definește acest gest. Acesta este folosit pentru a mișca imaginea la stânga. Practic a translata poziția imaginii inițiale la o alta poziție, la stânga față de centrul ecranului.

Mișcare la dreapta (gest de mișcare imagine)	- se folosește degetul arătător deschis, iar toate celelalte degete sunt închise/strânse în palmă. Mișcarea degetului la dreapta astfel încât acesta să deseneze virtual o linie definește acest gest. Acesta este folosit pentru a mișca imaginea la dreapta. Practic a translata poziția imaginii inițiale la o altă poziție, la dreapta față de centrul ecranului.
Înapoi (gest de navigare înapoi)	- toate degetele de la mână sunt implicate în realizarea acestui gest. Acesta presupune mișcarea/glisarea mâinii deschise/degetele mâinii întinse de la stânga la dreapta. Palma mâinii trebuie sa fie poziționată perpendicular față de planul LM. Acest gest este folosit pentru a trece la imaginea precedentă.
Înainte (gest de navigare înainte)	- toate degetele de la mână sunt implicate în realizarea acestui gest. Acesta presupune mișcarea/glisarea mâinii deschise/degetele mâinii întinse de la dreapta la stânga. Palma mâinii trebuie sa fie poziționata perpendicular fata de planul LM. Acest gest este folosit pentru a trece la imaginea următoarea.
Observație: Cele două gesturi se pot definii ca flexia (înapoi) și extensia (înainte) palmei, orientate perpendicular pe planul LM.	

Tabelul 2.1.1. Gesturi definite și descrierea lor

Din tabelul de mai sus se pot observa următoarele: 10 din cele 11 gesturi sunt perechi, care funcționează complementar unul față de celălalt. Fiecare gest dezvoltat este făcut cu ajutorul unei singure mâini și acesta este dinamic, implică o acțiune definită de timp și spațiu. 3 perechi de gesturi dezvoltate (observațiile după fiecare pereche de gesturi) se aseamănă între ele întrucât implică flexia și extensia degetelor/mâinii/palmei. Aceste 3 perechi de gesturi putând fi utilizate și în alte scopuri cum ar fi recuperarea medicală (capitol de care vom discuta în următoarele secțiuni), în special a articulațiilor mâinilor.

Aceiași cercetători în anul 2022 [BHI22] propun un algoritm bazat pe Fisher-HHT pentru detectarea gesturilor dinamice din baza de date LeapGestureDB [2AME20]. Aceștia au obținut performanțe pe algoritmul propus împreună cu clasificatorul KNN și SVM de 81.45% și respectiv 87.47%.

Modul de definire al gesturilor discutate mai sus este realizat prin colectarea datelor direct oferite de LM. Acestea se referă la: poziția palmei și poziția vârfurilor degetelor. Aceste 6 poziții se colectează în spațiul 3D pe cele 3 axe x, y si z. Astfel în baza de date care stă la baza acestor gesturi sunt colectate 18 tipuri de date. Colectarea datelor a fost realizată pe un eșantion de 10 persoane (3 bărbați și 7 femei), dintre care doar o persoană este stângace. Fiecare persoană a repetat gestul de 5 ori și s-a ajuns la 11x10x5=550 de date în baza de date. Pentru clasificare acestor gesturi s-a folosit un SVM (Suport Vector Machine) și s-a obținut o acuratețe de recunoaștere a gesturilor de 81%, iar în anul 2020 aceștia au reușit să crească performanța modelului la 91.73%. Cel mai mic scor a fost înregistrat la recunoașterea gestului de rotire la stânga și la dreapta (50% și 60%) întrucât se face de multe ori

confuzia între cele două mâini. S-a observat în literatura de specialitate că cele mai mari probleme de detecție pentru senzorul LM sunt înregistrate când este vorba de gesturi care implică rotirea mâinii și/sau rotirea degetelor.

O soluție pentru eliminarea acestei probleme este tratată în articolul [POL18]. Acesta vine cu o soluție hardware prin folosirea a doi senzori LM pe un stativ perpendiculari unul față de altul. Din punct de vedere al costurilor această soluție nu este optimă. În plus ca și dotări tehnologice este nevoie de o putere de calcul ridicată pentru PC-ul care conectează cei doi senzori si procesează datele venite de la aceștia.

Tot o soluție ce implică utilizarea a două dispozitive hardware cu scopul de a elimina confuziile ce apar la executarea gesturilor este discutată în [CHE19]. Dispozitivul Mayo Armband [MAYO23] este utilizat pentru a detecta activitatea musculară din brațul ce execută gestul. Pentru ca gestul să fie executat cu succes trebuie ca să se detecteze mișcare de dispozitivul LM pe una dintre mâini, iar mâna care se mișcă să fie și ea detectată/validată prin activitatea musculară detectă de MAYO.

O altă soluție ce implică utilizarea unui produs software de captare a gesturilor este discutată în [LI22]. Aici se folosește pe lângă dispozitivul LM și software-ul MAYA [SHU19] cu ajutorul căreia se poate detecta mișcarea. Astfel dacă gestul este făcut și se detectează mișcare se execută gestul, atunci se poate clasifica ca gest corect. Cercetătorii din [LI22] au dezvoltat 8 gesturi care pot fi aplicate pe aplicații educaționale. Astfel pentru crearea lor au folosit aproximativ 200000 de date, iar clasificarea s-a realizat printr-un model de rețea neuronală cu învățare profundă (DNN).

Direcția gestului este importantă în a fi luată în considerare pentru a crește acuratețea în clasificare a acestuia. Astfel în [ERD16] se propune un sistem de prezicere a obiectelor ce vor fii selectate prin utilizarea unor tehnici de ML. Precizia de prezicere a obiectului ce a va fii selectat a fost de 95%. Aceste performanțe sunt înregistrate după experimentarea de 20 de ori a gestului.

Numeroase articole ce tratează recunoașterea gesturilor alfa-numerice au fost găsite în literatura de specialitate. Detectarea acestora se face cu ajutorul mai multor dispozitive cum ar fi: camera foto/video, LM, Microsoft Kinect, mănuși speciale construite [SHI20], etc. Față de gesturile folosite pentru înlocuirea anumitor acțiuni gesturile alfa-numerice sunt gesturi statice, detectarea lor făcându-se prin interpretarea unei anumite stări a mâinii. Principalele articole care tratează acest caz au în componența titlurilor cuvintele *limbajul semnelor americ*ane (ASL). În continuare vor fi exemplificate câteva din lucrările științifice care tratează această temă de limbajul semnelor americane. Se va pune accentual pe prezentarea gesturilor create, precum și pe performanțele de detectare a acestora.

Un studiu din anul 2014 al [CHU14] sunt utilizate două modele de clasificare KNN (K-nearest neighbors) și SVM în care s-a obținut pe detectarea gesturilor alfabetice o acuratețe de 72.78% și 79.83%. Aceste performanțe scăzute sunt normale, dacă stăm să ne gândim că în urmă cu 10 ani nu se dispunea de baze de date mari pentru gesturi statice și de putere de calcul ridicată.

Cel mai simplu mod de a detecta un gest ASL este atunci când acesta este static. În [NAG16] se propune un model MLP (MultiLayer Perceptron) pentru recunoaște 26 de gesturi statice ce reprezintă cele 26 de litere din alfabet. Astfel s-a generat un set de date de 520 de rânduri cu 20 de caracteristici ce descriu gesturile

alfabetice. În figura 2.1.1 se pot observa cum arată gesturile statice și ce literă este atribuită fiecăruia. Modelul MLP a înregistrat o rată de recunoaștere a gesturilor de 96.15%.

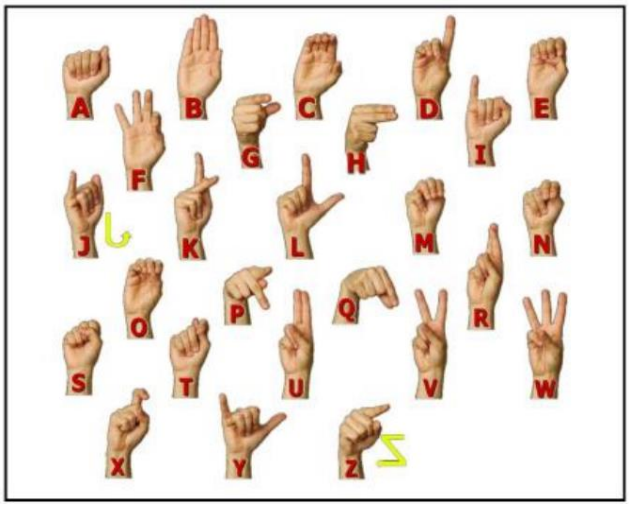

Figura 2.1.1. Gesturile statice alfabetice [NAG16]

Limbajul semnelor americane este discutat în [CHO18]. Aici se propune un algoritm care să poată recunoaște 7 litere din alfabet (A, E, M, N, O, S, T). Dacă privim la figura 2.1.1 observăm că gesturile statice asociate acestor litere sunt asemănătoare. Astfel cercetătorii au propus un algoritm bazat pe formule matematice, care la testare a obținut o acurateță medie de recunoaștere a celor 7 gesturi egală cu 96.1%.

În [WAN17] și [WAN18] se propun gesturi statice și dinamice pentru recunoașterea cifrelor. Astfel pentru prima cercetare s-au obținut performanțe de 95% legate de rata de recunoaștere a gesturilor, iar în a doua cercetare s-a obținut 93.1% în numerotarea dinamică. Performanța mai scăzută este normală să apară la a doua cercetare, deoarece s-a încercat crearea de gesturi dinamice, care să numeroteze/cronometreze prin gesturi cifrele. Trecerea de la o cifră la alta se face dinamic, iar aceasta aduce scăderea performanței în recunoaștere a gestului.

În anul 2019 se propune o rețea neuronală recurentă folosită pentru recunoașterea ASL. Astfel în [AVO19] s-a atins o precizie în recunoașterea gesturilor alfa-numerice de 96%. Modelul de rețea este antrenat pe un set de date ce descrie gesturile bazate pe unghiuri formate între degete și pe unghiuri formate între degete și mână. RNN a fost antrenat și pe un set de date de la SHREC (Shape Retrieval Contest) [SME17], iar acuratețea acestuia a crescut la 96.41%. În figura 2.1.2 este prezentat cum pot fii confundate gesturile între ele.

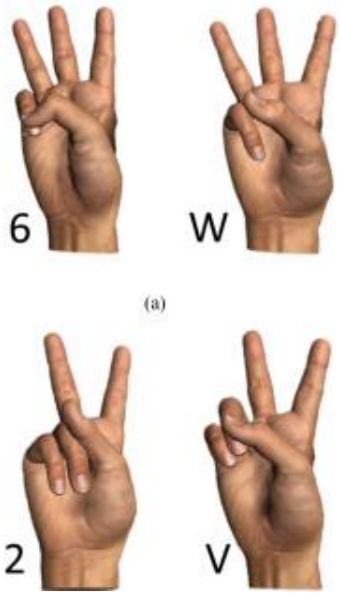

Figura 2.1.2. Confundarea gesturilor statice ASL [AVO19]

Tot pentru interpretarea limbajului semnelor americane (ASL), în [JEN21] propun un set de date pentru antrenarea unei rețele neuronale artificiale (ANN). Astfel datele ce descriu gesturile ASL sunt: poziția mâinii stângi și drepte, poziția degetelor de la mâna stângă și dreaptă, viteza de mișcare a mâinii stângi și drepte, rotirea, înclinarea și grația mâinilor, 10 date de tip boolean care semnifică dacă degetele sunt întinse (1) sau închise (0) în palmă și distanța dintre mâini. Gesturile au fost create tot pentru LM, iar acuratețea înregistrată a fost de 98%. Pentru a vizualiza cum a fost proiectată ANN se poate urmării figura 2.1.2. Aici se poate vizualiza stratul ascuns (stratul roșu), stratul de intrare reprezentat de caracteristicile rețelei și stratul de ieșire (stratul galben). Fiecare nod din rețea este conectat cu fiecare nod din stratul următor.

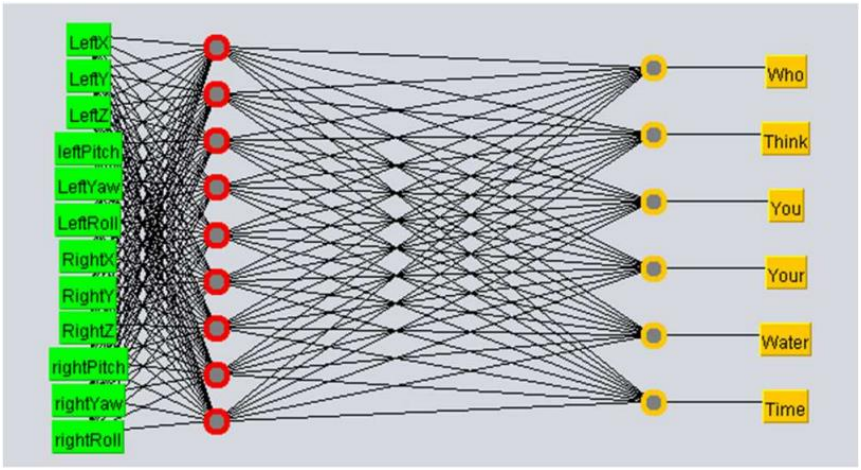

Figura 2.1.2. Exemplu al ANN utilizate [JEN21]

În [SAA22] se discută despre folosirea unei baze de date ce descrie 30 de gesturi [FIL18]. Astfel sunt clasificate prin metode de clasificare bazate pe rețele neuronale recurente (RNN) gesturi dinamice și statice. Sunt folosite 2 clasificatoare Bidirectional Recurrent Unit (Bi-GRU)si Bidirectional Long Short Term Memory (Bi-LSTM), iar performanțele obținute sunt de 96.97% și respectiv 97.28% acurateţe a gesturilor în figura 2.1.3 se poate observa care este cadrul (framework-ul) modelelor folosite.

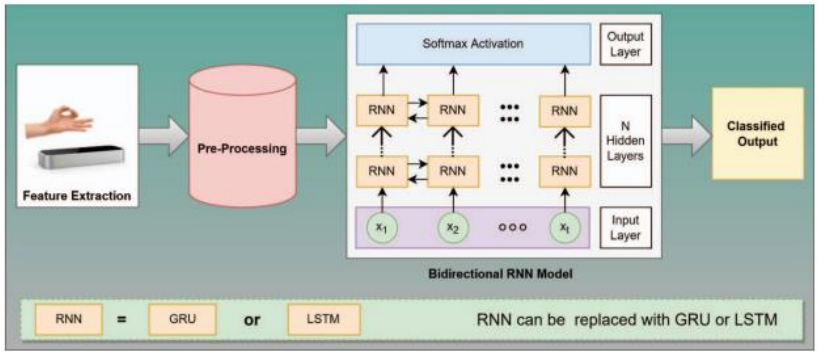

Figura 2.1.3. Framework-ul RNN – GRU/LSTM [SAA22]

Dacă privim spre performanțele obținute de cercetătorii care au avut interes în clasificarea gesturilor statice alfa-numerice (ASL) observăm că, în fiecare an s-au făcut progrese, iar precizia de recunoaștere a gesturilor este tot mai mare.

Gesturile au aplicabilitate nu doar în scrierea de litere sau cifre, ci și în recuperarea medicală. Astfel numeroși cercetători au utilizat dispozitivul LM și au construit gesturi atât dinamice cât și statice, care pot fii utilizate în recuperarea mobilității mâinilor. În zilele noastre sunt folosite frecvent gesturile reabilitarea fiziologică a mâinilor [KAV20], [RAK20].

În anul 2016 un grup de cercetători au folosit dispozitivul LM și tehnologia VR pentru recuperarea mobilității mâinilor [LUP16]. Aceștia propun un joc în care se fac exerciții bazate pe gesturi în VR. Tot bazat pe realitatea virtuală [CAR20] propun o aplicație de detectare a gesturilor mâinilor care să fie utilizată în recuperare. Utilizarea de algoritmi matematici în aplicații de realitate augumentată controlată prin gesturi LM este propusă în [XUE18]. Algoritmul propus de recunoaștere al gesturilor făcute în recuperarea mobilității mâinilor a atins performanțe egale cu 98.9%. Realitatea augumentată și gesturile LM sunt discutate și în [HUO21]. Cercetătorii acestui articol propun 3 modele de clasificare a gesturilor LM: SVM, KNN și DNN. Aceste modele sunt folosite pentru interpretarea gesturilor statice care semnifică următoarele acțiuni sau stări: victorie, ok, aprobare, învins și sună-mă. Forma celor 5 gesturi poate fi observată în figura 2.1.4. Gesturile clasificate pot fii utilizate în interacțiunea/traducerea cuvintelor spuse de persoanele surdo-mute. Pe cele 4 modele s-au obținut următoarele acurateți: 98% KNN, 97% DNN și peste 98% SVM.

Wei X. și restul [WEI17] propun o metodă bazată pe gesturi pentru antrenare, astfel încât să se poată face recuperarea după un accident vascular cerebral (AVC).

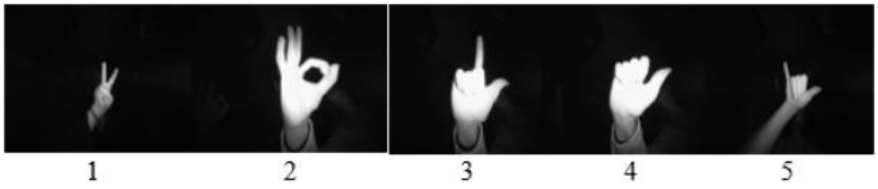

Figura 2.1.4. Forma gesturilor statice [HUO21]

Gesturile recunoscute de LM implică atingerea pe rând a degetelor între ele. Au fost definit 8 combinații de gesturi, pentru care s-a înregistrat o acuratețe de clasificare de 84.5%. La acest studiu au participat 198 de persoane. În anul 2017 se propun 7 gesturi care sunt utilizate în recuperarea mâinilor: flexia și extensia mâinii, deschiderea și închiderea degetelor, atingerea degetelor, apăsarea degetelor, rotirea palmei, extensia și flexia degetelor [LI18] și mișcarea radial-ulnar. [LI17] propun două modele SVM și KNN pentru clasificarea acestor gesturi. S-au obținut acurateți egale cu 97.29% pentru SVM și 97.71% pentru KNN. În figura 2.1.5 sunt prezentate gesturile create pentru recuperare. Alte modele de clasificare a gesturilor de recuperare sunt propuse în [WAL19]. Aici se folosesc modelele LDA și SVM pentru clasificarea gestului de flexie și extensie al mâinii. Cercetătorii au obținut o acuratețe medie pentru cele două modele de 88%.

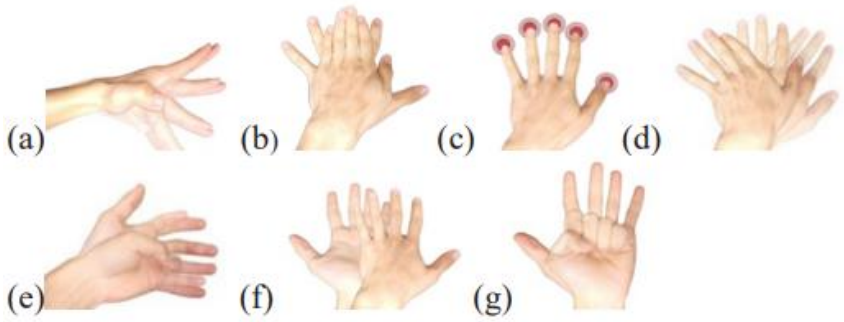

Figura 2.1.5. Gesturile create pentru recuperare [LI17]

În anul 2020 gesturile LM sunt aplicate în jocuri piatră, hârtie, foarfece. [BRO20] și [YAN20] propun utilizarea modelelor RF (Random Forest), Naive Bayes (NB), Ada Boost (AB) și respectiv DNN pentru clasificarea celor 3 gesturi. Astfel primii au obținut rezultate la acuratețea modelelor de: 99.3%. 99.2 % și 96.8%. Modelul DNN a înregistrat o acuratețe de 98% pe un set de date ce descriu cele 3 gesturi de 30000. Parametrii luați în considerare în descrierea gesturilor au fost: poziția vârfului degetelor și joncțiunile oaselor fiecărui deget. Acest lucru a fost posibil datorită capacității LM de a face diferența dintre degete și joncțiunile acestora. Aceiași parametrii sunt luați în considerare în descrierea gesturilor LM și în [Li20]. Aici se propune un algoritm de potrivire fuzzy (SFM) care a atins performanțe între 94%-100% pentru gesturile statice și 90% pentru gesturile dinamice.

Dacă mai sus am discutat despre folosirea gesturilor în jocuri reale, mai departe voi discuta despre folosire gesturilor LM în jocuri 3D. Acestea au ca scop principal creșterea interactivității om-calculator. Gesturile LM sunt folosite în jocuri pentru a controla obiecte 3D și/sau animații. În [DZI18] se folosesc gesturile pentru controlul animațiilor caracterului. S-a utilizat metoda de propagare înapoi a unei rețele neuronale (BPNN) pentru clasificarea gesturilor. Astfel s-a obținut 96.7% acuratețe pentru modelul BPNN. Pentru gesturile predefinite LM și utilizate în explorarea unor spații 3D, în [SUM19] s-a utilizat un model KNN și a înregistrat acuratețe de 99.72%. Tot pentru controlul modelelor 3D în [KRI19] se propune un model de rețea neuronală profundă pentru recunoașterea gesturilor dinamice. Acest model a atins o rată de recunoaștere a gesturilor egală cu 94%. În figura 2.1.6. se arată care sunt gesturile dinamice clasificate de DNN. Au fost definite 7 gesturi. Fiecare dintre gest reprezintă o acțiune pe care utilizatorul o face pentru a controla un pian virtual. Gesturile au fost descrise fiecare de 1019 date. Acestea se referă la pozițiile degetelor, unghiurile dintre palmă și pianul virtual, rotirea, înclinarea și grația palmei. Un model DNN a fost aplicat pentru clasificarea gesturilor ce reprezintă scrierea unui număr în aer. În [BAS20] s-au utilizat rețele neuronale LSTM și TCM (Temporal Convolutional Network) pentru clasificarea a 10 gesturi. Modelele au fost antrenate pe un set de 1200 de date, obținându-se acuratețea de 99.5% pentru LSTM și 97% pentru TCM. Tot pentru detectarea scrierii în aer în lucrarea [KUM17] sunt detectate cuvintele. Astfel gesturile care simulează scrierea de cuvinte sunt clasificate prin diferite metode. Modelul Markov ascuns (HMM) și modelul BLASTM-Nn sunt utilizate pentru clasificarea gaturilor de scriere a cuvintelor. Setul de date cuprinde 560 de fraze desenate, iar modelul HMM a înregistrat o acuratețe de 81.25%, în timp ce modelul BLASTM-NN a înregistrat o acuratețe de 86.88%.

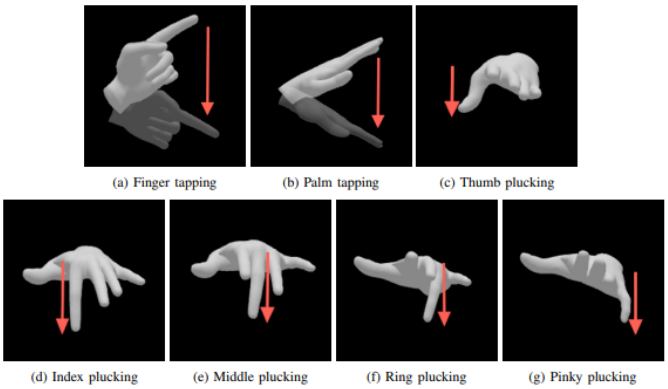

Figura 2.1.6. Gesturile dinamice pentru controlul 3D al unui pian virtual [KRI19]

Gesturile pot fi utilizate și în detectarea sau prevenția unor anumite boli. Astefl în [1MOS20] și [2MOS20] gesturile LM sunt utilizate pentru detectare boli Parkinson. Acest grup de cercetători Folosesc modele de clasificare ca: SVM, DT, RF și KNN pentru clasificarea gesturilor: deget apăsat, pronația și supinația mâinii, închiderea și deschiderea mâinii. Cele 4 modele au fost antrenate pe un set de date ce descriu gesturile, iar performanțele lor sunt următoarele: 85.1% pentru SVM, 78.9% pentru DT, 81.5% pentru RF și 98.4% pentru KNN.

Gesturile statice sunt cele mai ușor de procesat, acestea nu depind de variabila timp. În [BAN21] se propune un model de rețea neuronală convoluțională pentru clasificare 10 gesturi statice. Fiecare gest este descris de 2000 de imagini. În figura 2.1.7 se poate observa cum arată fiecare imagine pe cele 10 gesturi. La procesarea acestora se observă o zonă mai luminoasă la partea mâinilor. Fiind vorba de gesturi statice, s-a înregistrat și aici o performanță ridicată la clasificare: 99.4%.

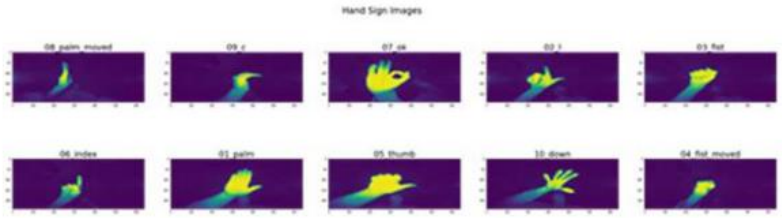

Figura 2.1.7. Imagini cu cele 10 gesturi statice [BAN21]

În anul 2023 K. Sneha și R. Kayalvizhi propun un o rețea neuronală cu clasificatorul KNN pentru clasificarea gesturilor alfa-numerice [SME23]. Aceste gesturi fiind utilizate de persoanele paralizate fiind un mod de comunicare și interacționare între ele și persoanele din jur. Pentru modelul KNN s-a obținut o acuratețe de 96.13%.

În acest subcapitol am observat cum pot fi clasificate gesturile prin diferite modele de clasificare. Totodată am observat unde pot fi ele utilizate dar și cât de diversificate sunt. În tabelul 2.1.2 se face o scurtă trecere în revistă asupra celor mai importante cercetări despre care am discutat.

Nr. Crt.	Autori	Gest	Modele/ Clasificatoare/ Testare clasică	Acurateţe/ Performanţe/ Rata de recunoaştere
1	[CHU14]	26 gesturi statice pentru literele alfabetului (ASL)	KNN și SVM	72.78% și 79.83
2	[ERD16]	Gest detectare intenției	Tehnici de ML	95% predicție
3	[NAG16]	26 gesturi statice pentru literele alfabetului (ASL)	MLP	96.15%
4	[AME16]	Gesturi din baze de date LM	Clasificare valori gest	81%
5	[WEI17]	8 gesturi care implică atingerea degetelor de la mână între ele	testare prin 198 de participanți	84.50%
6	[LI17]	7 gesturi: flexie și extensie, deschidere și închidere degete, atingere degete, apăsare degete, rotire palmă, extensia și flexia degetelor și mișcare radial-ulnar	SVM și KNN	97.29% și 97.71%
7	[WAN17]	10 gesturi statice ce exprimă cifre	Metodă deterministă – algoritmi matematici	95%
8	[KUM17]	Gesturi pentru scrierea de cuvinte	HMM și BLSTM-NN	81.25% și 86.88%
9	ROS17	Gesturi statice alfanumerice sau de punctuație	CNN	90% precizie de recunoaștere
10	[WAN18]	12 gesturi statice	CNN	93.10%
11	[CHO18]	Gesturi statice pentru 7 litere din alfabet	CNN	96.10%
12	[DZI18]	Gesturi predefinite LM	BM	96.70%
13	[XUE18]	Gesturi de rotire a mâinii	Metodă deterministă – algoritmi matematici	98.8
14	[CHO18]	Geturi dinamice folosite în sala de operație	SVM și NB	99.58% și 98.74%
15	[KRI19]	Gesturi pentru controlul modelelor virtuale 3D	CNN	94%
16	[AVO19]	Gesturi alfanumerice (ASL)	RNN	96.41%

17	[WAL19]	Gestul dinamic de flexie și extensie	LDA și SVM	88%
18	[SUM19]	Gesturi predefinite LM	KNN	99.72%
19	[AME20]	Gesturi din baza de date LeapGestureDB	SVM	91.73%
20	[BAS20]	10 gesturi dinamice ce semnifică 10 cifre	CNN, LSTM și TCN	98.5%, 99.5% și 97%
21	[BRO20]	3 gesturi statice ce semnifică piatră, foarfece, hârtie	RF, NB și AB	99.3%, 99.2% și 96.8%
22	[LI20]	Gesturi de mișcare a mâinilor	SFM algoritm de potrivire fuzzy	94%-100% gesturi statice, 90% gesturi dinamice
23	[1MOS20]	3 gesturi de mișcare: apăsare cu degetul, pronație și supinație, închiderea și deschiderea mâinii	SVM, DT și RF	85.1%, 78.9% și 81.5
24	[2MOS20]	3 gesturi de mișcare: apăsare cu degetul, pronație și supinație, închiderea și deschiderea mâinii	KNN	98.40%
25	[YAN20]	3 gesturi statice ce semnifică piatră, foarfece, hârtie	DNN	98%
26	[JEN21]	26 gesturi statice pentru literele alfabetului (ASL)	ANN	98%
27	[HUO21]	5 gesturi statice ce exprimă acțiuni	KNN, DNN și SVM	98%, 97% si 98%
28	[BAN21]	10 gesturi statice	CNN	99.40%
29	[RAK21]	Gesturi dinamice de semnătură	SVM, RF și NB	74% precizie de recunoaștere
30	[SAA22]	Dinamice și statice definite în baza de date ASL [FIL18]	Bi-GRU și Bi-LSTM	96.97% și 97.28%
31	[BHI22]	Gesturi din baza de date LeapGestureDB [AME20]	KNN-Fisher-HHT și SVM-Fisher-HHT	81.45% și 87.47%

32	[SNE23]	Gesturi statice alfa-numerice	KNN	96.13%

Tabelul 2.1.2. Reflectarea rezultatelor cercetărilor

2.2. Starea actuală în domeniul realității virtuale (VR), augumentate (AR) și mixte (MX)

Folosirea noilor concepte precum VR, AR și MR împreună cu gesturile sau dispozitivele de interacțiune în aplicații 3D crește interactivitatea dintre om și calculator (HCI). Utilizarea acestor concepte duce la apariția unor aplicații care sunt: robuste, sigure, repetabile și rentabile [RAD21]. Astfel în acest subcapitol voi discuta pe scurt care sunt tendințele în ceea ce privește utilizarea VR, AR sau MR în literatura de specialitate. Ultimele cercetări [CHE17] și [BIR18] arată că numărul de aplicații din domeniul educațional/formării medicale și numărul de lucrări științifice scrise au crescut în ultimii 20 de ani.

Realitatea virtuală (VR) este utilizată în diferite domenii cum ar fi: medical, industrial, ingineresc. În [CHH19], [PIN22] și [CHA22] se propun aplicații 3D pentru planificarea intervențiilor chirurgicale. Astfel studenții de la medicină pot să simuleze intervenții chirurgicale pentru a repeta sau a se perfecționa pe o anumită procedură. Simularea se realizează cu ajutorul unor ochelari speciali, HTC Vive [HTCV23].

O altă procedură pe care studenții de la medicină o pot antrena prin VR este cea de resuscitare (RCP) [EVE22]. Principalul avantaj al utilizării tehnologiei VR în medicină/educare/antrenare este acela că procesul poate fi repetat fără a pune viața în pericol a pacientului.

Pregătirea și instruirea studenților de la medicină pe anumite proceduri ce trebuie urmate înaintea unei operații este discutată în [GRU21]. În figura 2.2.1 se pot observa pașii pe care studenții de la medicină îi urmează într-o aplicație bazată pe VR. În aplicație studenții parcurg etapele de pregătire pentru sala de operație: îmbrăcarea hainelor chirurgicale, efectuarea dezinfecției mâinilor, ajungerea în sala de operație și rezistarea la o perioadă lungă de nemișcare și necontaminare, spălarea pregătitoare a unui loc unde se va executa intervenția chirurgicală. În această aplicație studenții sunt ghidați asupra etapelor pe care trebuie să le efectueze. Practic această aplicație ține cont de un principiu destul de des utilizat în aplicații educaționale și anume gamificarea [LAV19].

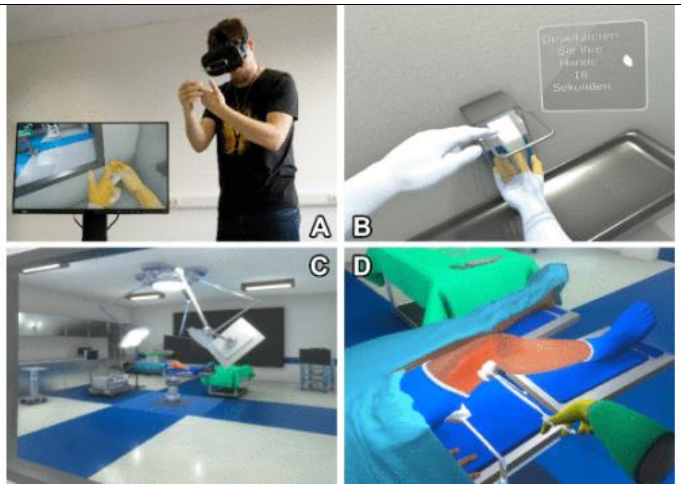

Figura 2.2.1. Etapele de pregătire pentru o intervenție chirurgicală în VR [GRU21]

Tot o structurare pe etape din ceea ce privește învățarea în domeniul medical utilizând VR este propusă în [RAJ19]. Principalul scop al aplicației propuse "AirwayVR" este de antrenare și formarea abilităților de intubare. În figura 2.2.2 se pot vedea pașii pe care utilizatorii îi fac pentru intubare. Beneficiul dat de această aplicație este constituit de posibilitatea de antrenare într-un mod cât mai apropiat de realitate a studenților de la medicină. Aceștia nu pot să exerseze un astfel de procedeu pe oameni sănătoși, iar când sunt puși să o facă aceștia execută procedeul doar pe oameni ce au nevoi de acest lucru. Prin afișajele de pe monitorul ochelarilor HTC Vive studenții află ce trebuie să facă în procedura de intubare.

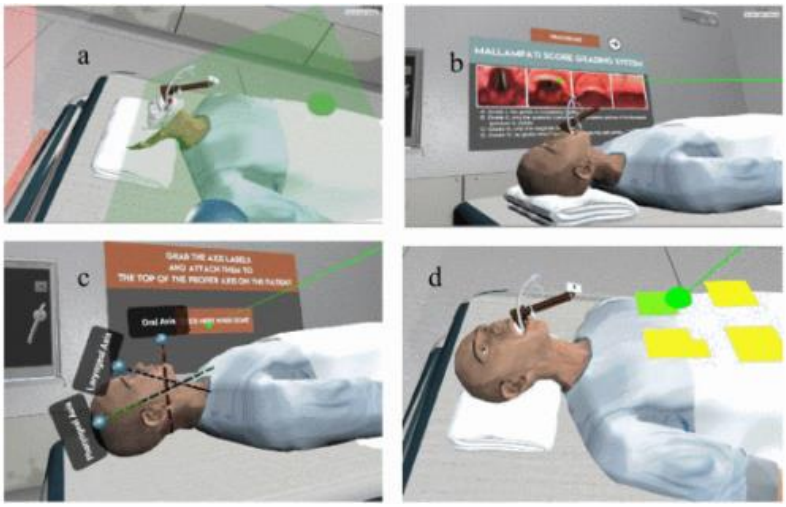

Figura 2.2.1. Etapele de intubare a unui pacient virtual [RAJ19]

Așa cum am discutat în subcapitolul precedent, gesturile pot fi utilizate în procesul de reabilitare/recuperare. VR poate să sprijine acest lucru prin apropierea care se face utilizând diferite echipamente/ochelari VR. Astfel în [WU19] și [GEO21] se propun aplicații care să ajute utilizatorii în procesul de recuperare. În aceste aplicații se oferă prin diferite mijloace cum trebuie să fie făcute exercițiile de recuperare, dar și dacă exercițiile făcute sunt corecte. În plus se pot oferii medii virtuale care să simuleze locuința pacienților, astfel încât aceștia la întoarcerea acasă să poată să se descurce cu mișcarea prin locuință. Aceste medii virtuale ajută în special persoanele care au suferit un AVC (atac vascular cerebral) [Nag21].

Un alt domeniu în care VR are aplicabilitate este industria. În [LIU20] și [JIA21] se propun aplicații care să ajute angajații din mediul industrial sau inginerești să execute anumiți pași. Inginerii electrici pot folosii VR pentru vizualizarea și asamblarea unui transformator.

Așa cum am menționat mai sus pe lângă VR avem și AR și MR. Tehnologia AR derivă din VR, dar aici lumea virtuală cu cea reală se combină [TON20]. În AR lumea virtuală se suprapune/combină cu lumea reală cu ajutorul unui tag sa a unei etichete pe care dezvoltatorul de aplicații AR o stabilește. Pentru a stoca astfel de etichete este nevoie de baze de date ce permit acest lucru. Cea mai cunoscută bază de date utilizată în AR împreună cu motoarele de dezvoltare de jocuri/aplicații 3D Unity [UNIT23] este Vuforia [VUFO23].

La fel ca și AR, MR derivă din ar, această tehnologie fiind o continuare a AR. Dacă la AR se folosesc etichete pentru combinarea lumii reale cu cea virtuale, la MR acest lucru nu mai este necesar. Astfel se pot combina cele două lumi fără a ține cont de anumite etichete. Partea virtuală poate să apară în partea reală oriunde [MOA23].

Domeniile de aplicabilitate ale acestor două tehnologii sunt: educație [SCH22], medicină [SCH22], [MOR18], imagistică [POP20] – partea de explorare a imaginilor medicale și industrie [WAN19].

Pentru a avea o privire de ansamblu asupra: domeniului, tehnologiilor utilizate și scopul cercetărilor bazate pe VR, Ar și MR în tabelul 2.2.1 sunt afișate aceste lucruri identificate.

Nr. Crt.	**Autori**	**Domeniu**	**Tehnologii și unelte**	**Scop**
1	[MOR18]	Medical	AR, Unity, ochelari Hololens, Vuforia	Asistarea personalului din domeniul medical în executarea unor intervenții.
2	[WAN19]	Industrial	MR, Unity, ochelari HTC Vive	Antrenarea angajaților din industria de producție.
3	[RAJ19]	Medical	AR, Unity, ochelari HTC Vive	Antrenarea studenților de la medicină pentru executarea unei proceduri de intubare.

4	[CHH19]	Medical	VR, Unity, ochelari HTC Vive	Antrenarea studenților de la medicină pentru executarea unei intervenții chirurgicale.
5	[WU19]	Medical	VR, Unity, ochelari HTC Vive	Recuperarea mobilității membrelor superioare.
6	[PIN20]	Medical	VR, Unity, ochelari HTC Vive	Antrenarea studenților și medicilor rezidenți pentru efectuarea intervențiilor chirurgicale.
7	[LIU20]	Industrial	VR, Unity, ochelari Oculus VR	Instruirea angajaților din domeniul industrial.
8	[POP20]	Medical	AR și MR, Unity, Vuforia, ochelarii MagicLeap și Hololens	Oferirea către medici a unei experiențe vizuale adecvate pentru mai multe imagini medicale.
9	[GEO21]	Medical	VR, Unity, ochelarii Oculus Quest 2	Recuperarea la domiciliu și oferirea unui antrenor virtual ce arată exercițiile.
10	[NAG21]	Medical	VR, Unity, căști HMD	Antrenarea persoanelor care au suferit un AVC.
11	[GRU21]	Medical	VR, Unity, ochelari HTC Vive	Îmbunătățirea pregătirii studenților de la medicină din ceea ce privește etapele care trebuie făcute în sala de operație.
12	[JIA21]	Inginerie	VR, Unity, 3D MAX, ochelari Oculus VR	Instruirea inginerilor din domeniul electric.
13	[SCH22]	Medical	MR, ochelari HTC Vive	Antrenarea paramedicilor și asistentelor medicale de urgență pe un scenariu cu un pacient care a suferit arsuri.
14	[HEN22]	Educație IT	MR, Unity, ochelari HTC Vive	Învățarea studenților de la o universitate tehnică.
15	[EVE22]	Medical	VR și AR, ochelari VR și AR	Antrenarea cadrelor medicale și a studenților pentru efectuarea RCP.

16	[CHA22]	Medical	VR, ochelari VR	Antrenarea studenților de la medicină pentru practica clinică.

2.3. Concluzii și perspective după studierea cercetărilor din domeniu

În acest capitol am văzut care este stadiul în literatura de specialitate din punct de vedere al dezvoltării de gesturi pentru dispozitivul Leap Motion. În plus s-au identificat principalele gesturi construite pentru acest dispozitiv și cum pot fi ele clasificate. Domeniile de aplicabilitate a gesturilor dar și modul lor de clasificare rămâne unul deschis, în special în ceea ce privește construirea de gesturi dinamice care implică o abordare mai complexă. Folosirea gesturilor dar și a tehnologiilor bazate pe VR, AR sau MR duce la un progres ridicat din punct de vedere al interactivității om-calculator.

Principalele contribuții aduse de mine în domeniul Calculatoare și Tehnologia Informației din acest capitol sunt date de structurarea și analiza gesturilor care reiese din tabelul 2.1.2 și analiza modelelor de clasificare a gesturilor. O altă contribuție este reprezentată de identificarea principalelor aplicații ce folosesc tehnologii de VR, AR sau MR și care împreună cu gesturile LM ajută la creșterea interactivității om-calculator. Structurarea pe domeniu de aplicabilitate și tehnologii folosite pentru VR, AR și MR reprezintă și ea o contribuție în domeniu Calculatoare și Tehnologia Informației.

După analiza/studiul domeniului problemei în următorul capitol voi prezenta bazele teoretice care stau la construirea de gesturi și aplicații VR, AR și MR.

3. DEFINIREA ȘI CLASIFICAREA GESTURILOR RECUNOSCUTE DE LEAP MOTION

În acest capitol se vor prezenta principalii pași care trebuie făcuți din punct de vedere tehnic pentru a crea un gest dinamic pentru dispozitivul Leap Motion (LM).

Deoarece în această teză s-a pus accent pe clasificarea corectă și precisă a gesturilor, voi prezenta baza teoretică destinată modelelor de clasificare pentru gesturile LM. În plus pentru a putea măsura calitatea clasificării gesturilor voi prezenta și cum se calculează metricile care fac referire la: precizie, acuratețe, recall și f1 scor ale modelelor de clasificare. Totodată voi arăta cum se completează o matrice de tranziție pentru observarea corectă a rezultatelor modelelor de clasificare.

În continuare voi enumera principalele subcapitole ale acestei părți din teza de doctorat:

- Definirea gesturilor LM
- Definirea modelelor de clasificare a gesturilor
- Prezentarea metricilor modelelor de clasificare
- Prezentarea concluzii privitoare la bazele teoretice necesare pentru definirea și clasificarea gesturilor

3.1. Definirea unui gest LM

Dispozitivul Leap Motion este echipat cu două camere monocromatice și trei led-uri cu infraroșu. Aceste camere pot face până la 300 de cadre pe secundă, performanțele lor depinzând de PC-ul/laptopul la care LM este conectat. În medie un PC utilizat în comun de utilizatori, înregistrează 120 de cadre pe secundă. Software-ul acestui dispozitiv poate să înregistreze de la ce mână este utilizată, până la diferențierea fiecărui deget de la mână. În plus acesta oferă cu precizie pozițiile mâinilor dar și ale degetelor în spațiul 3D. 4 gesturi predefinite prezintă dispozitivul LM: Screen_Tap, Key_Tap, Swipe și Circle. Pentru a definii noi gesturi pe care dispozitivul LM să le recunoască se pot utiliza pozițiile degetelor și/sau ale mâinilor [NIC16].

În Figura 3.1 este exemplificat care sunt pașii necesari pentru crearea unui gest. 4 blocuri decizionale sunt prezente în această schemă. Pentru început se face o verificare a conectivității dispozitivului LM la PC/Laptop, dacă acesta este conectat se trece mai departe, iar dacă nu se reia verificarea. Dacă acesta este conectat se verifică prezența mâinilor în cadrele înregistrate, iar dacă nu sunt prezente, atunci se reia verificarea. După verificarea pozitivă a prezenței mâinilor se inițializează poziția mijlocului palmei ale mâinilor prezente, apoi se verifică prezența degetelor în cadre. Dacă degetul/degetele pe care dorim să le căutăm sunt prezente, se inițializează poziția vârfului acestora. Dacă acestea nu sunt prezente se reia căutarea lor. După obținerea celor necesare: poziție degete și palmă, se pot face anumite calcule care au

rol în definirea gestului pe care dorim să-l construim: distanțe, unghiuri de rotire, semi-perimetre, direcții. Ultima verificare este destinată validării variabilelor ce definesc gestul construit. Dacă acele variabile sunt în intervalul de valori experimentat, atunci se inițiază/creează gestul. Intervalele de valori sunt stabilite experimental, acestea depinzând de mărimea mâinilor virtuale folosite în definirea gesturilor. Mâinile virtuale 3D vin împreună cu pachetul software Leap Motion destinat utilizării în editorul Unity. Aceste mâini primesc pozițiile reale ale mâinilor și ale degetelor față de dispozitivul LM, apoi acestea se mișcă copiind mișcarea mâinilor reale.

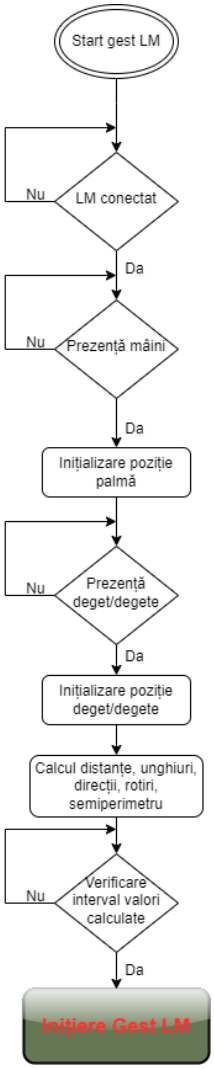

Figura 3.1. Construirea gesturilor LM

3.2. Modele folosite în clasificarea gesturilor

Pentru a face o clasificare riguroasă a gesturilor definite, am utilizat 12 modele de clasificare fiecare dintre aceste modele având caracteristici specifice acestora. În continuare voi enumera modelele, iar mai apoi voi explica fiecare model în parte:

- Logistic Regression – LR (regresie logistică)
- Linear Discriminant Analysis - LDA (Analiză discriminantă liniară)
- Kneighbors Classifier – KNN (Clasificator k de vecini)
- Decision Tree Classifier - CART (Clasificator de arbore de decizie)
- Gaussian Naive Bayes – NB (Clasificator Gaussian Naive Bayes)
- SVC1 (Clasificator suport vector 1) - SVC
- SVC2 (Clasificator suport vector 2) - Liniar SVM
- SVM (Mașină vectorială de suport) - RBF SVM
- Random Forest Classifier (Clasificator de arbori aleatoriu) - Random Forest
- Ada Boost Classifier (Clasificator Ada Boost) - AdaBoost
- MLPClassifier (Clasificator de perceptron multistrat) – MLP
- DNNClassifier (Clasificator cu model de rețea neuronală cu învățare profundă)

Primele 11 modele au fost folosite pe o rețea neuronală din biblioteca sklearn, iar ultimul model pe o rețea neuronală din biblioteca tensorflow. Toate scripturile aplicate sunt scripturi Python.

Logistic regression – LR (regresia logistică)

Regresia logistică este o tehnică de analiză statistică folosită pentru a analiza probabilitatea de aparținere a unui set de date la una sau mai multe clase. Acest model este cel mai des utilizat pentru clasificări binare, dar pot fi utilizate și pe clasificări de tipul multi-clasă.

Modelul poate fi exprimat matematic printr-o funcție logaritmică a raportului dintre probabilitatea de apartenență la o clasă și probabilitatea de apartenență la alte clase. Acest raport se transformă printr-o funcție sigmoidă pentru ca probabilitățile să fie tot timpul între 0 și 1 [THO17].

Exemplu:

Să presupunem că avem un set de date care sunt etichetate ca aparținătoare uneia din 6 clase: A, B, C, D, E și F. Pentru a utiliza LR pentru clasificare se va antrena pe rând fiecare dintre cele 6 clase. Î timpul antrenării fiecare model logistic va fi instruit să clasifice datele că aparțin sau nu uneia dintre clase. După antrenare se face o predicție pe un set de date de testare, și se vor obține 6 probabilități de apartenență

la fiecare clasă. Clasa cu cea mai mare probabilitate, adică clasa a cărei probabilități este cea mai aproape de 1, va fi considerată clasa prezisă.

Dacă probabilitățile sunt dispuse în felul următor: 0.3 pentru clasa A, 0.1 pentru clasa B, 0.3 pentru clasa C, 0.8 pentru clasa D, 0.7 pentru clasa E și 0.5 pentru clasa F, atunci clasa prezisă este clasa D.

Linear Discriminant Analysis - LDA (Analiză discriminantă liniară)

Analiza discriminantă liniară este o tehnică de clasificare utilizată pentru a găsii combinații lineare de variabile independente care să permită separarea/identificarea claselor identificând un set de variabile independente. Practic se face o identificare de caracteristici care sunt aparținătoare doar unei clase. Modelul LDA presupune că datele sunt distribuite în mode normal în fiecare clasă și că acestea au covarianțe egale. Modelul se bazează pe găsirea unei suprafețe plane de n-1 dimensiuni într-un spațiu de n dimensiuni. Practic acest model reduce dimensiunea cu 1 față de dimensiunea inițială [CHA20].

Exemplu:

Fie 3 clase care descriu nivelul unui gest folosit în recuperarea mobilității mâinilor. Fiecare clasă este descrisă de **n** caracteristici: poziții degete, poziție palmă, distanțe degete – mijlocul palmei, semi perimetrul format de figura geometrică cu puncte în vârful degetelor mare, mijlociu și mijlocul palmei.

Pentru a utiliza LDA se va estima media și matricea de covarianță a fiecărei dintre cele 3 clase. Aceste date se utilizează pentru calcularea coeficienților pentru reducerea dimensiunii spațiului numit și hiperplan. După calcularea coeficienților de separare, se pot clasifica nivelurile gestului pe baza poziției nivelurilor față de această suprafață. Clasificarea se face pe baza semnului valorii calculului discriminant. Dacă valoarea discriminantului este mai mare decât 0 atunci gestul este clasificat unui anume nivel din cele 3, iar dacă este negativă gestul nu face parte din nici un nivel/clasă.

Kneighbors Classifier – KNN (Clasificator k de vecini)

Modelul KNN se bazează pe identificarea celor K cele mai apropiate puncte de date în setul de antrenament pentru a determina clasa unui nou set de date. KNN poate fi utilizat pentru clasificarea unui număr finit de clase. Acest algoritm nu necesită etapă de antrenare propriu zisă . Se bazează prin învățarea prin asociere și stabilește clasa unui exemplu de teste pe baza similarității acestuia cu K exemple.

Pașii algoritmului KNN:

- Stabilirea valorii K, numărul de vecini cel mai apropiat considerat
- Măsurarea distanței euclidiene între noua instanță și toate punctele din setul de antrenament
- Stabilirea celor K puncte (vecinii cei mai apropiați) cele mai apropiate de noua instanță de date
- Clasificarea noii instanțe în funcție de clasa majoritară a acelor vecini [WAN23]

Exemplu:

Fie un gest care vrem să-l clasificăm că este prezent sau nu (gestul este împărțit în două clase 1 – gest prezent și 0 gest absent). Gestul este descris din 3 caracteristici: poziție palmă, unghi de rotire al mâinii pe axa y și direcția degetului mare al mâinii.

Pentru a clasifica gestul utilizând KNN, îi dăm variabilei K valoarea 3. Se măsoară distanța euclidiană între noua instanță de date ce descriu noul gest și toate celelalte gesturi din setul de antrenament. Cele mai apropiate 3 gesturi vor fi considerate vecinii cei mai apropiați față de noul gest introdus.

După identificarea celor mai apropiate gesturi, se clasifică gestul nou în funcție de clasa majoritară. Dacă 2 dintre cei 3 vecini sunt identificați cu 1, adică fac parte din gest și unul este identificat cu 0, adică nu face parte din gest, atunci noul gest introdus va fi identificat cu valoarea 1, adică va face parte din gest.

Decision Tree Classifier - CART (Clasificator de arbore de decizie)

CART este un model de clasificare și regresie bazat pe arbori de decizie. Acesta construiește arbori de decizie prin împărțirea setului de date în funcție de caracteristici/parametrii pentru a obține grupuri omogene de date în ceea ce privește clasificarea. Practic împarte un grup de date/caracteristici în subgrupuri cu scop de a obține subgrupuri omogene pentru clasificare.

Când modelul CART este utilizat pentru clasificare, acesta alege caracteristica care separă datele în două subgrupuri omogene. Acesta utilizează impuritatea Gini sau entropia informațională. Impuritatea Gini calculează probabilitatea greșelii atunci când se prezice clasa unei date aleatorii din setul de date. Următorul pas al modelului CART este împărțirea subgrupurilor obținute la pasul precedent prin aceeași metodă. Acest proces se repetă până când nu mai este posibilă împărțirea în subgrupuri sau se atinge un prag predefinit împărțiri. Astfel la terminarea împărțirilor va rezulta o clasă țintă [OOI17].

Exemplu:

Fie un set de date care descriu un gest care simulează prinderea unui obiect (1) sau lăsarea acestuia (0) din mână. Caracteristicile gestului sunt: *semi-perimetrul* format de figura geometrică cu puncte în vârful degetului mare, mijlociu și mijlocul palmei și *distanța* mâinii față de obiectul ce trebuie prins.

Modelul CART va face o primă alegere pentru a separa datele. Astfel primul nivel de alegere este valoarea semi-perimetrului. Dacă acest este mai mic de 100 de unități atunci va considera că gestul este prezent, iar dacă este mai mare va considera că gestul nu este prezent. Următorul nivel de clasificare î-l reprezintă valoarea distanței dintre mâna ce este folosită și obiectul ce trebuie prins. Dacă această distanță este mai mică de 1 față de obiect, atunci se va considera că obiectul este prins. În Figura 3.2 este expus în mod grafic cum este construit arborele de decizie CART pentru acest exemplu.

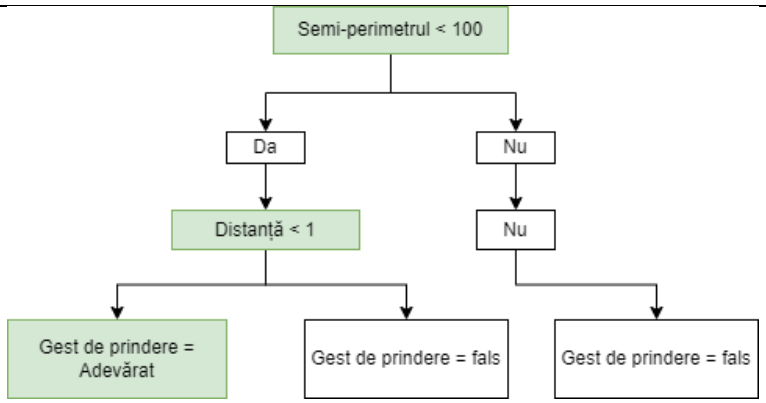

Figura 3.2.1 Exemplu arborele de decizie CART

Gaussian Naive Bayes – NB (Clasificator Gaussian Naive Bayes)

Clasificatorul NB este un model de clasificare probabilistică care se bazează pe teorema lui Bayes. Acesta presupune că variabilele de intrare sunt independente între ele (nu depind unele de altele), ceea ce în realitate nu este totdeauna adevărat. Această presupunere/simplificare a modelului îi oferă o mare eficiență computațională.

Pentru a face o predicție prin NB, se calculează probabilitatea ca un anume set de date, aparține unei anumite clase. Aici se ține cont de valorile atribuite acelui set de date. Modelul NB presupune că fiecare atribut este independent de celelalte și se calculează probabilitățile condiționate ale claselor [VIJ23].

Exemplu:

Fie un gest ce descrie prinderea sau lăsarea unui obiect. Se dorește clasificarea acestuia ca fiind prezent (1) sau absent (0). Caracteristicile ce descriu gestul sunt următoarele: poziția degetelor folosite, distanța dintre mijlocul palmei și obiectul ce trebuie prins, semi-perimetrul format de figura geometrică cu puncte în vârful degetelor mare, mijlociu și mijlocul palmei. Pornind de la un set de date cu atributele/caracteristicile de mai sus, modelul NB poate fi antrenat pentru a face o predicție pentru un set de date ce descrie gestul în funcție de valorile caracteristicilor.

Mai precis, pentru un gest de prindere descris, modelul NB va calcula probabilitatea condiționată a fiecărei clase. Ținând cont de valorile caracteristicilor. Clasa cu valoarea probabilității condiționată maximă, va fi considerată că este clasa prezisă pentru acel gest de prindere.

SVC1 (Clasificator suport vector 1) - SVC

SVC este un model de învățare automată care este utilizat pentru clasificări și regresii. Acesta este o metodă supervizată care încearcă să găsească un hiperplan care să separe datele în clasele dorite a fi clusterizate. Scopul împărțirii datelor în hiperplan este de a maximiza distanța dintre hiperplan și cele mai apropiate puncte din fiecare clasă. Distanța este numită și marginea de separare și este maximizată pentru a asigura o bună generalizare la noi date.

SVC poate fi utilizat și în clasificarea mai multor clase prin folosirea unui algoritm numit OvA (One-vs-All) sau OvO (One-vs-One). În OvA, modelul SVC construiește un clasificator binar pentru fiecare clasă, iar în OvO, SVC construiește un clasificator binar pentru fiecare pereche de clase posibile.

Exemplu:

Fie un gest ce exprimă flexia și extensia mâinii. Acesta este împărțit în 6 niveluri. Caracteristicile gestului fiind: tip mână – stânga (0) sau dreapta (1), poziție mijloc palmă în spațiul 3D, unghiuri de rotire, înclinare și grație ale mâinii și vectorii direcție a mâinii și a degetului mare. Se dorește clasificarea nivelurilor gestului de flexie și extensie. Modelul SVC primește fiecare caracteristică a gestului ca dată de intrare. Acesta este antrenat pentru a recunoaște unul dintre cele 3 niveluri ale gestului pentru a le clasifica corect. Modelul se asigură să găsească un hiperplan pentru a separa datele. Se calculează distanțele dintre date, apoi se clasifică nivelurile gestului în funcție de valoarea distanțelor.

SVC2 (Clasificator suport vector 2) - Liniar SVM

Modelul Mașină cu Vector Suport (SVM) liniar este o versiune a algoritmului SVM în care se construiește un hiperplan liniar cu rol în separarea datelor claselor. Acest hiperplan este construit astfel încât să maximizeze distanța dintre el și cele mai apropiate puncte din fiecare clasă (numite și vectori suport). Distanța se numește marginea de separare (granița de separare) și este maximizată pentru a asigura o bună generalizare la noi date.

Pentru construirea unui model SVM liniar trebuie ca datele de antrenare să fie reprezentate de vectori de caracteristici numerice și datele să fie etichetate cu clasa corespunzătoare.

Exemplu:

Dacă se dorește clasificarea unui gest ce exprimă mișcarea de pronație și supinație a mâinii folosind modelul SVM liniar pentru început trebuie identificate caracteristicile acestuia. Caracteristicile gestului: poziția mijlocului palmei în spațiul 3D, unghiul de rotire al mâinii pe axa x, vectorul direcție al degetului mare. După identificarea caracteristicilor se etichetează fiecare grup de caracteristici punând o clasă care corespunde lor. Consider 1 pentru datele care identifică gestul de pronație și supinație (clasa 1) și 0 pentru datele ce nu identifică gestul (clasa 2).

Modelul SVM liniar se va antrena pe datele din setul de date generat, apoi acesta va construii un hiperplan care va separa cele două clase. În timpul antrenării modelul va ajusta hiperplanul pentru maximizarea marginii de separare dintre cele două clase.

SVM (Mașină vectorială de suport) - RBF SVM

Mașina Vectorială de Suport cu funcție radială (RBF-SVM) este un alt tip de model SVM. Spre deosebire de celelalte modele de SVM menționate aceasta folosește o funcție de kernel non-liniară pentru transformarea datelor de intrare într-un spațiu de dimensiune superioară. Acest model folosește la fel ca și celelalte modele de SVM un hiperplan pentru separarea datelor.

Datele de intrare trebuie să fie bazate pe vectori de caracteristici numerice pentru a construi un model RBF SVM. Toate datele de intrare sunt etichetate cu una dinte clasele dorite în funcție de caracteristicile lor [NAS17].

Exemplu:

Fie un gest ce exprimă mișcarea de flexie și extensie a mâinii. Gestul este descris de următoarele caracteristici: poziție mijloc palmă în spațiul 3D, unghiul de rotire al mâini pe axa y, vectorii direcție al mâinii și al degetului mare. Datele sunt etichetate ca aparținătoare (clasa 1) sau nu (clasa 2) a gestului de flexie și extensie. Caracteristicile gestului reprezintă datele de antrenare pentru modelul RBF SVM. Acesta va construii un hiperplan non-liniar care va avea rolul de separare a datelor în funcție de cele două clase. În timpul antrenării se vor face ajustări parametrilor hiperplanului de separare și a funcției kernel RBF pentru a maximiza precizia de clasificare.

Cele trei modele SVM sunt cel mai des utilizate în clasificări binare, dar acestea pot fi utilizate și în clasificări multiple, dacă se ține cont că datele pot fi clasificate în două clase mari, iar mai departe aceste clase să fie și ele la rândul lor în subclase.

Random Forest Classifier (Clasificator de arbori aleatoriu)

Modelul Random Forest este un clasificator care combină mai mulți arbori de decizie pentru a îmbunătății precizia și stabilitatea predicțiilor. Acesta este un ansamblu de arbori de decizie, în care fiecare arbore este antrenat pe un subset aleatoriu de date cu un subset aleatoriu de caracteristici.

Procesul de antrenare al modelului Random Forest începe prin selectarea unui număr aleatoriu de exemple și a unui număr aleatoriu de caracteristici din setul de date inițial. În pasul următor se construiește un arbore de decizie pe baza celor două: exemple și caracteristici. Procesul este repetat de mai multe ori astfel încât să fie dezvoltați mai mulți arbori de decizie independenți.

Fiecare arbore de decizie din Random Forest este utilizat pentru a face o predicție individuală. Clasa cu cea mai mare frecvență de apariție este aleasă ca fiind clasa finală [KHA19].

Exemplu:

Fie gestul de flexie și extensie al mâinii. Acesta este descris de unghiul de rotire al mâinii pe axa y, poziția mâinii/palmei și vectorul direcție al degetului mare. Dacă se pornește de la un set de date etichetate ca aparținătoare gestului de flexie (1) și extensie (0), modelul Random Forest poate fi antrenat pentru a face predicții precise pentru noi date neetichetate ca aparținătoare unuia dintre gesturile de flexie și extensie în funcție de valorile caracteristicilor menționate.

De exemplu dacă un set de date este compus din 1000 de rânduri, practic 1000 de valori ce descriu cele 3 caracteristici ale gestului de flexie și extensie, 300 dintre rânduri sunt etichetate cu 1 (gestul de flexie) și 700 cu 0 (gestul de extensie) și se dorește să se clasifice un nou rând ca aparținător al uneia dintre clase. Modelul Random Forest va face o predicție individuală pe fiecare arbore de decizie, iar clasa cu cea mai mare apariție, clasa cu cea mai mare frecvență în apariție, va fi selectată ca fiind clasa finală pentru acel rând cu date ce descrie gestul.

Ada Boost Classifier (Clasificator Ada Boost)

Modelul Ada Boost este un model de învățare automată care folosește mai mulți clasificatori slabi (clasificatori construiți pe baza caracteristicilor din setul de date ce descriu gestul) pentru a obține o clasificare precisă. Un prim pas pe care algoritmul Ada Boost î-l face pentru clasificare este antrenarea unui clasificator slab pe întreg

setul de date, iar apoi ajustează ponderile exemplelor din setul de date pentru a se referii la exemplele care au fost clasificate greșit. Acest proces se repetă pentru fiecare clasificator slab adăugat, astfel încât modelele ulterioare se concentrează pe exemplele ce au fost greșit clasificate.

În timpul etapei de antrenare fiecare exemplu din setul de date primește o greutate inițială egală, iar după ce unul dintre clasificatorii slabi este antrenat, greutatea fiecărui exemplu este ajustată în funcție de performanța modelului. Greutatea mai mare este atribuită exemplelor ce au fost clasificate greșit, iar următorul clasificator slab se va orienta în funcție de exemplele clasificate greșit la pasul precedent.

Antrenarea modelului Ada Boost continuă prin adăugarea de clasificatori slabi succesivi, iar acesta este antrenat să se concentreze pe exemple care au fost clasificate greșit de clasificatorii anteriori. Modelul Ada Boost a fost construit pe baza conceptului "învață din greșeli și nu le mai repeta în viitor". Practic cu cât sunt clasificate mai multe exemple greșit de clasificatorii slabi, cu atât mai mult ultimul model ce combină rezultatele clasificatorilor slabi va fi mai puternic, având o performanță de clasificare corectă ridicată.

Exemplu:

Fie un gest care exprimă gestul de flexie și extensie al mâinii. Un set de date este format cu date care exprimă atât gestul de flexie cât și gestul de extensie. Gestul de flexie și extensie este descris în setul de date de următoarele caracteristici: poziție palmă, vector direcție al degetului mare și unghiul de rotire al mâinii pe axa y. Pentru început se antrenează întreg setul de date pentru primul clasificator slab (poziția mâinii). După prima antrenare toate datele care au fost clasificate greșit sunt luate în considerare și se adaugă următorul clasificator slab (vectorul direcție al degetului mare). Procesul este repetat pe setul de date rămase care au fost clasificate greșit și se adaugă următoarea caracteristică, unghiul de rotire. Datele sunt antrenate pe noul clasificator, iar la sfârșit clasificatoarele sunt combinate pentru a obține un clasificator puternic care poate detecta gestul de flexie și extensie cu o precizie ridicată. Ponderile datelor din setul de date se modifică în funcție de clasificarea greșită la fiecare adăugare de clasificator slab.

MLPClassifier – MLP (Clasificator de perceptron multistrat)

MLP este un model de rețea neuronală artificială care este utilizată pentru probleme de clasificare și regresie. Modelul este format din mai multe straturi de neuroni, în care fiecare strat este conectat la mai multe straturi adiacente. Neuronii dintr-un strat sunt conectați la neuronii din straturile adiacente, iar legăturile sunt ponderate și ajustate prin procesul de antrenare. Numărul de straturi ascunse din rețeaua neuronală cu modelul MLP poate fi ajustat, la fel ca și numărul de iterații, în funcție de complexitatea datelor de intrare ce descriu clasele dorite a fi clasificate.

Ultimul strat de neuroni din rețea se numește stratul de ieșire, iar acesta produce o valoare de ieșire pentru fiecare clasă posibilă. Valorile de ieșire sunt trecute print funcția *softmax()* pentru a obține probabilități pentru fiecare clasă. Clasa cu cea mai mare probabilitate va fi aleasă ca și clasă prezisă [ALM20].

Exemplu:

Fie un gest de rotire a mâinii (mișcarea de pronație și supinație) împărțit pe 3 nivele. Caracteristicile ce descriu gestul fiind: poziția mijlocului palmei în spațiul 3D,

vectorul direcție al mâinii și al degetului mare, unghiul de rotire al mâinii pe axa x. Modelul MLP primește ca și date de intrare caracteristicile gestului. Acesta este antrenat pe setul de date pentru a recunoaște nivelurile 1, 2 sau 3 ale gestului. Ponderile conexiunilor dintre neuroni sunt ajustate în timpul antrenării. Scopul ajustării fiind minimizarea erorii de clasificare. După antrenarea modelului MLP poate fi utilizat pentru a face clasificările gestului de pronație și supinație al mâinii.

DNNClassifier – DNN (Clasificator cu model de rețea neuronală cu învățare profundă)

Deep Neural Network (DNN) este un model de rețea neuronală formată din mai multe straturi de neuroni. DNN este utilizat în probleme de clasificare și regresie, iar numărul de straturi poate să fie modificat în funcție de complexitatea problemei de clasificare.

Primul strat de neuroni se numește stratul de intrare. Caracteristicile claselor sunt scrise în acest strat. Următoarele straturi sunt straturile ascunse, a căror număr se poate calcula în funcție de setul de date de antrenare, de stratul de intrare și de stratul de ieșire.

$$N_h = \frac{N_s}{\alpha(N_i + N_o)}$$

Unde:

N_h – numărul de neuroni ascunși

N_i – numărul de intrări

N_o – numărul de ieșiri

N_s – numărul de date din setul de date

α – număr arbitrar ales între 2 și 10.

Ultimul strat de neuroni este stratul de ieșire, acesta produce o valoare de ieșire pentru fiecare clasă posibilă. Valorile de ieșire sunt trecute print funcția *softmax()* pentru a obține probabilități pentru fiecare clasă. Clasa cu cea mai mare probabilitate va fi aleasă ca și clasă prezisă [CHI19].

Exemplu:

Fie un set de date ce descriu gestul de flexie și extensie al mâinii. Caracteristicile gestului fiind: index mână (1- dreapta, 0 stânga), unghiul de rotire al mâinii, unghiul de înclinare al mâinii, ungiul de grație (înclinarea mâinii pe axa y), poziția mijlocului palmei în spațiul 3D, vectorii direcție al mâinii și al degetului mare în spațiul 3D. Datele de intrare ale modelului DNN sunt caracteristicile gestului. Numărul de date din setul de date pentru gestul de flexie și extensie este 969. Gestul este împărțit în 6 nivele. Câte 3 nivele pentru fiecare mână. Se dorește clasificarea cât mai bună a gestului de flexie și extensie.

Cansiderăm un număr de două straturi ascunse cu 17 neuroni ascunși pe fiecare strat. Numărul de neuroni ascunși fiind calculați după formula:

$$N_h = \frac{N_s}{\alpha(N_i + N_o)} = \frac{969}{3(13+6)} = \frac{969}{57} = 17$$

Modelul DNN este antrenat pentru a recunoaște fiecare dinte cele 6 nivele/clase ale gestului de flexie și extensie. O exprimare grafică a rețelei neuronale cu modelul DNN este afișată în figura 3.2.2. Aici se pot observa numărul de straturi ascunse dar și straturile de intrare și ieșire ale rețelei.

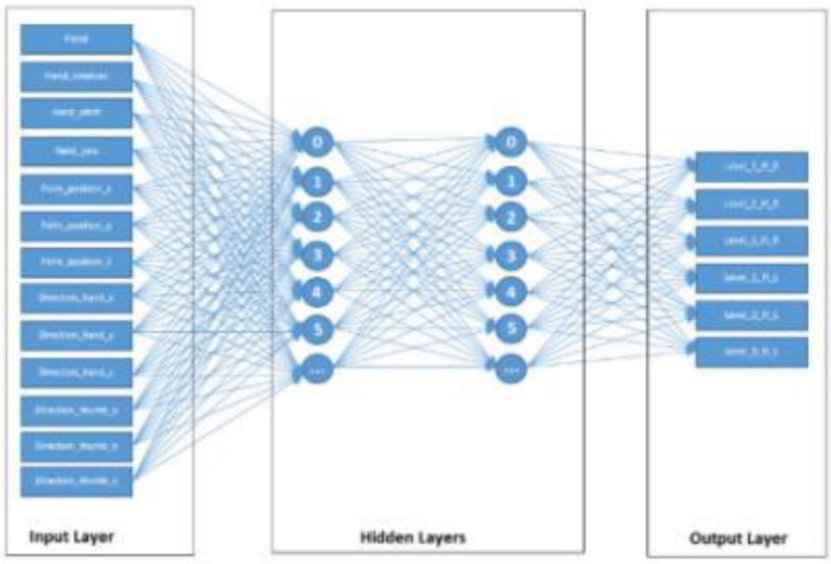

Figura 3.2.2. Rețeaua neuronală rezultată – modelul DNN [1NIC20]

3.3. Metrici ale modelelor de clasificare a gesturilor

Pentru a măsura eficiența modelelor de clasificare a gesturilor menționate în subcapitolul 3.2, în continuare voi definii principalii termeni folosiți:

- Adevărat pozitiv
- Adevărat negativ
- Fals pozitiv
- Fals negativ
- Precizia
- Funcția recall
- Funcția f1-score
- Acuratețea
- Matricea de confuzie

Toți termenii definiți vor fi explicați și printr-un exemplu pentru a fixa mai bine modul în care aceștia sunt folosiți.

Adevărat pozitiv (AP)

Este detectat atunci când un nivel al unui gestului este clasificat corect ca fiind parte din acel nivel.

Exemplu:

Fie un gest împărțit pe 6 niveluri. Pentru un total de 500 de cazuri de testare, 450 dintre teste au fost clasificate corect ca făcând parte din unul dintre niveluri.

Adevărat negativ (AN)

Este detectat atunci când un nivel al gestului este clasificat corect că nu face parte din acel nivel.

Exemplu:

Fie un gest împărțit pe 6 niveluri. Pentru un total de 500 de cazuri de testare, 30 dintre teste au fost clasificate corect ca nefăcând parte din unul dintre niveluri.

Fals pozitiv (FP)

Este detectat atunci când un nivel al gestului este clasificat greșit ca fiind parte din acel nivel.

Exemplu:

Pentru un număr de total de 500 de cazuri de testare al unui gest 10 dintre acestea au fost clasificate greșit ca făcând parte din unul dintre nivelurile gestului, dar acesta aparținea altui nivel. Acesta definește confuziile care apar între niveluri; Spre exemplu nivelul 0 cu nivelul 1.

Fals negativ (FN)

Este detectat atunci când un nivel al gestului este clasificat greșit ca nefăcând parte din acel nivel.

Pentru un număr de total de 500 de cazuri de testare al unui gest 10 dintre acestea au fost clasificate greșit ca nefăcând parte din unul dintre nivelurile gestului. Cazuri în care unul dintre nivelurile gestului este prezis ca nefăcând parte din acesta.

Precizia (P)

Este definită de raportul dintre sumele cazurilor adevărat pozitive și sumele cazurilor adevărat pozitive și fals pozitive.

$$P = \frac{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}}}{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} FP\_nivel_{\boldsymbol{k}}}$$

Unde:

P – precizia

k – numărul curent al nivelului (clasei) ale gestului

n – numărul maxim de nivele ale gestului

AP_nivel – numărul de cazuri adevărat pozitive

FP_nivel – numărul de cazuri fals pozitive

Exemplu:

Fie un gest împărțit pe 6 niveluri și 500 de cazuri de testare. 450 dintre teste au fost clasificate corect ca făcând parte din unul dintre niveluri. Teste corecte fiind distribuite în mod egal pe fiecare nivel câte 75. Practic 75 dintre teste s-a clasificat corect fiecare nivel în parte. 10 teste din cele 500 au fost identificate ca fals pozitive.

Pentru datele menționate mai sus precizia se calculează în felul următor:

$$P = \frac{450}{450 + 10} = 0.978$$

Funcția Recall (R)

Este definită de raportul dintre sumele cazurilor adevărat pozitive și sumele cazurilor adevărat pozitive și fals negative.

$$R = \frac{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}}}{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} FN\_nivel_{\boldsymbol{k}}}$$

Unde:

R – funcția recall

k – numărul curent al nivelului (clasei) ale gestului

n – numărul maxim de nivele ale gestului

AP_nivel – numărul de cazuri adevărat pozitive

FN_nivel – numărul de cazuri fals negative

Exemplu:

Fie un gest împărțit pe 6 niveluri și 500 de cazuri de testare. 450 dintre teste au fost clasificate corect ca făcând parte din unul dintre niveluri, adevărat pozitive. Teste corecte fiind distribuite în mod egal pe fiecare nivel câte 75. Practic 75 dintre teste s-a clasificat corect fiecare nivel în parte. 10 teste din cele 500 au fost identificate ca fals negative.

Pentru datele menționate mai sus funcția recall se calculează în felul următor se calculează în felul următor:

$$R = \frac{450}{450 + 10} = 0.978$$

Funcția F1-score (F1)

Este calculat ca fiind raportul dintre dublul produsului dintre precizie și recall și suma dintre precizie și recall.

$$F1 = \frac{2 * P * R}{P + R}$$

Unde:

F1 – funcția f1 score

R – funcția recall

P – precizia

Exemplu:

Dacă considerăm precizia și funcția recall egale cu valorile calculate pentru acestea pe exemplul celor 500 de cazuri de testare, atunci funcția f1 score este egală cu:

$$F1 = \frac{2 * 0.978 * 0.978}{0.978 + 0.978} = 0.977$$

Acuratețea (A)

Este calculată ca fiind raportul dintre suma cazurilor adevărat pozitive și negative cu suma tuturor cazurilor (adevărat pozitive, negative și fals pozitive și negative).

$$A = \frac{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} AN\_nivel_{\boldsymbol{k}}}{\sum_{k=1}^{n} AP\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} AN\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} FP\_nivel_{\boldsymbol{k}} + \sum_{k=1}^{n} FN\_nivel_{\boldsymbol{k}}}$$

Unde:

A – acuratețea

k – numărul curent al nivelului (clasei) ale gestului

n – numărul maxim de nivele ale gestului

AP_nivel – numărul de cazuri adevărat pozitive

AN_nivel – numărul de cazuri adevărat negative

FP_nivel – numărul de cazuri fals pozitive

FN_nivel – numărul de cazuri fals negative

Exemplu:

Fie un gest împărțit pe 6 niveluri și 500 de cazuri de testare. 450 dintre teste au fost clasificate corect ca făcând parte din unul dintre niveluri. Teste corecte fiind distribuite în mod egal pe fiecare nivel câte 75. Practic 75 dintre teste s-a clasificat corect fiecare nivel în parte. 10 teste din cele 500 au fost identificate ca fals pozitive, 10 au fost fals negative și 30 adevărat negative.

Pentru datele menționate în acest exemplu acuratețea este egală cu:

$$A = \frac{480}{500} = 0.96$$

Matricea de confuzie

Matricea de confuzie este o exprimare matematică, un vector bidimensional în care se exprimă cazurile adevărat pozitive, adevărat negative, fals pozitive și fals negative într-un test de verificare a eficienței unui model de clasificare. O matrice de confuzie ideală are numere pozitive pe diagonala principală, iar pe celelalte elemente are 0. Asta înseamnă că toate cazurile de testare au fost adevărat pozitive, adică toate cazurile au fost identificate corec ca aparținătoare a unei clase.

Se formează o matrice pătratică de ordin egal cu numărul de clase de clasificare a gesturilor.

Dacă sunt **n** clase, atunci ordinul matricei este **n.**

Matricea se completează în felul următor: pe diagonala principală se pun cazurile adevărat pozitive, iar în celelalte elemente se găsesc cazurile care au fost detectate greșit.

În tabelul 3.1 este exemplificat cum se completează o matrice de confuzie pentru clasificarea unui gest împărțit pe 6 niveluri.

		Valori prezise					
		Nivel 0	Nivel 1	Nivel 2	Nivel 3	Nivel 4	Nivel 5
Valori actuale	Nivel 0	AP (0, 0)	(0, 1)	(0, 2)	(0, 3)	(0, 4)	(0, 5)
	Nivel 1	(1, 0)	AP (1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)
	Nivel 2	(2, 0)	(2, 1)	AP (2, 2)	(2, 3)	(2, 4)	(2, 5)
	Nivel 3	(3, 0)	(3, 1)	(3, 2)	AP (3, 3)	(3, 4)	(3, 5)
	Nivel 4	(4, 0)	(4, 1)	(4, 2)	(4, 3)	AP (4, 4)	(4, 4)
	Nivel 5	(5, 0)	(5, 1)	(5, 2)	(5, 3)	(5, 4)	AP (5, 5)

Tabelul 3.1 Exemplu matrice de confuzie pentru un gest pe 6 niveluri

În continuare voi explica cum se completează matricea de confuzie pentru nivelul 0:

- numărul de cazuri adevărat pozitive (AP) este dat de intersecția nivelului 0 al valorilor actuale cu nivelul 0 al valorilor prezise.
- numărul de cazuri fals negative (FN) este dat de suma elementelor ce se găsesc în continuarea liniei de unde apar cazurile adevărat pozitive
    - (0, 1) + (0, 2) + (0, 3) + (0, 4) + (0,5)
- numărul de cazuri fals pozitive (FP) este dat de suma elementelor ce se găsesc pe aceiași coloană cu cazurile adevărat pozitive, dare sunt sub acestea
    - (1, 0) + (2, 0) + (3, 0) + (4, 0) + (5, 0)
- suma restului elementelor din matrice dau numărul de cazuri adevărat negative (AN) pentru nivelul 0
    - (1, 1) + (1, 2) + (1, 3) + (1, 4) + (1, 5) + (2, 1) + (2, 2) + (2, 3) + (2, 4) + (2, 5) + (3, 1) + (3, 2) + (3, 3) + (3, 4) + (3, 5) + (4, 1) + (4, 2) + (4, 3) + (4, 4) + (4, 5) + (5, 1) + (5, 2) + (5, 3) + (5, 4) + (5, 5)

3.4. Concluzii

În acest capitol s-a arătat cum se poate defini un gest pentru care să fie recunoscut de dispozitivul Leap Motion. Pentru a face o clasificare a gesturilor definite precum și pentru a crește precizia și acuratețea de recunoaștere a gesturilor LM au fost prezentate principalele modele folosite în clasificare. Fiecare model a fost definit, dar a fost arătat și printr-un exemplu, pentru a fixa mai bine partea teoretică legată de acestea cu partea practică.

În continuarea acestui capitol au fost prezentați principalii termeni și principalele formule utilizate în analiza statistică. După definirea acestor termeni și ei au fost însoțiți de exemple.

Contribuțiile din domeniul Calculatoare și Tehnologia Informației ale acestui capitol sunt referite de modelul de construire a gestului, dar și de exemplele oferite pe cazuri reale în clasificare prin modele de clasificare a gesturilor. Partea teoretică din acest capitol stă la baza următoarelor capitole care fac referire la contribuțiile aduse de mine în domeniu Calculatoare și Tehnologia Informației.

În următoarele capitole vor fi prezentate toate contribuțiile mele din această teză de doctorat.

4. ALGORITMI DE DETECȚIE A GESTURILOR

În acest capitol se vor prezenta principalii algoritmi creați pentru detecția gesturilor folosind formule matematice de calcul al distanțelor dintre două puncte, semi perimetrelor sau unghiurilor de rotire. Vor fi explicate toate gesturile create precum și modul de utilizare împreună cu aplicabilitatea acestora. Gesturile create fac referire la: prinderea și lăsarea obiectelor 3D (3 moduri diferite de abordare pentru gestul de prindere), gestul de flexie și extensie a încheieturii mâinii, gestul de rotire al palmei în sensul acelor de ceasornic și invers și gestul de strângere și deschidere a mâinii. După cum se poate observa, ultimele trei gesturi enumerate fac referire la mișcări ample care sunt des utilizate în recuperarea mâinilor. În plus se va arăta cum se poate crea un gest complex pentru a putea fi detectat de LM (Leap Motion). Acest gest complex implică utilizarea ambelor mâini.

4.1. Algoritmul de detecție a gestului de prindere

Gestul de prindere poate să implice folosirea unor combinații mai mari de degete și să exprime aceiași acțiune. Am abordat 3 aspecte pentru detecția gestului de prindere:

- gestul de prindere a unui obiect cu implicare a degetului arătător și mare al mâinii în acțiune; acest gest fiind cel mai utilizat de oameni când vorbim de prinderea unui obiect mic cu forme regulate sau neregulate (exemplu: prinderea unui pix de pe masă implică folosirea degetului mare și a celui arătător de la mână).
- gestul de prindere al unui obiect prin implicarea degetului arătător în acțiune (gestul cu efect magnetic de prindere al obiectelor); chiar dacă în viața reală nu putem prinde obiecte prin folosirea unui singur deget, am ales să abordez folosirea acestuia deoarece de multe ori când arătăm spre ceva ne referim de fapt ca dorim să-l prindem sau să aducem mâna spre acesta.
- gestul de prindere complex care implică folosirea degetului mare al mâinii si oricare deget/degete de la mâna; prin abordarea acestui gest putem să acoperim orice caz când ne referim la prinderea unui obiect de forme regulate sau neregulate cu diferite dimensiuni.

Dimensiunea mâinii virtuale folosite pentru construirea acestor gesturi este egală cu 25 de unități în spațiul 3D Unity pentru toate cele 3 axe. Astfel la această dimensiune ne raportăm când vorbim de intervalele de valori generate de cele 3 gesturi.

Pentru cele trei abordări ale gestului de prindere menționate mai sus voi folosi următoarea notare: Gest de prindere 1, Gest de prindere 2 și Gest de prindere 3.

4.1.1. Gest de prindere 1

Acest gest este definit prin cele două degete de la mână: arătătorul și degetul mare. Poate exprima mai multe acțiuni cum ar fi: prinderea unui obiect de dimensiuni regulate sau neregulate, ciupirea obiectelor, dar și atingerea acestora. Gestul este realizat prin mișcarea de apropiere și depărtare a degetului arătător față de cel mare de la aceiași mână.

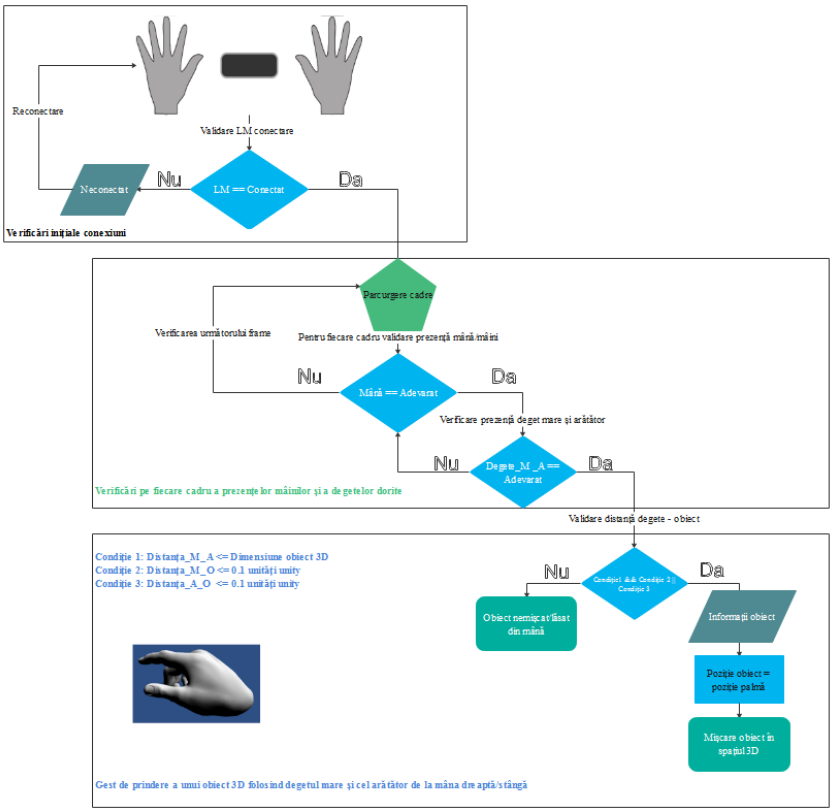

Figura 4.1.1.1

În Figura 4.1.1.1, primul modul face referire la verificarea conectivității dispozitivului LM la PC/laptop. În cazul în care dispozitivul nu este conectat se reîncearcă conectarea acestuia. Dacă acesta este conectat se trece la modulul al doilea de verificare a prezenței mâinilor. Aici se parcurge fiecare cadru pentru verificarea prezenței mâinilor, apoi daca mâna este prezentă se face validarea prezenței degetelor arătător și mare de la mână în cadru. Acestea pot fi distinse în fiecare *frame* nefăcându-se confuzia cu alte degete de la mână. În cazul în care mâna/mâinile nu sunt prezente în cadru, se face din nou verificarea prezenței acestora, la fel procedându-se și la prezența degetelor arătător și mare de la mână.

Ultimul pas din acest algoritm este cel mai complex. Acesta implică verificarea a trei distanțe:

- distanța dintre degetul mare și arătător (dist_1);
- distanța dintre degetul mare și obiectul ce trebuie prins (dist_2);
- distanța dintre degetul arătător și obiectul ce trebuie prins (dist_3).

Pentru a putea prinde un obiect 3D în mână se exemplifică următoarele condiții:

- *dist_1 <= dimensiunea obiectului 3D* (aria suprafeței de care se prinde mâna)
- *dist_2 <= 0.1 unități în spațiul 3D Unity*
- *dist_3 <= 0.1 unități în spațiul 3D Unity*

Pentru ca ultimele două condiții să poată fi respectate, dimensiunea/scala mâinilor utilizate din pachetul software LM trebuie să fie cel puțin egală cu 25 unități în spațiul 3D Unity pe fiecare dintre cele 3 axe (x, y, z).

Din cele trei inecuații descrise se poate spune că este prezentat un interval de valori cuprins între (0; 0.1] pentru ultimele două inecuații și (0; dimensiune obiect] pentru prima inecuație. Aceste trei intervale marchează faptul că gestul de prindere construit este unul dinamic, fiind realizat pe un interval de valori definit, nu pe o singură valoare.

Dacă condițiile 1, 2 și 3 sunt respectate, atunci obiectul poate fi prins și mișcat prin spațiul 3D.

În practică acest lucru se datorează posibilității de a atribui coordonatelor obiectului ce trebuie prins, coordonatele mâinii (coordonatele palmei mâinii) ce face gestul de prindere. După ce obiectul a fost prins prin gestul de prindere, se pot atribui efecte speciale în spațiul 3D pentru a spori interactivitatea dintre utilizator și computer cum ar fi:

- informarea utilizatorului despre numele obiectului prins;
- colorarea obiectului astfel încât utilizatorul să observe selecția obiectului;
- mărirea și micșorarea obiectului/deformarea acestuia pentru un efect cât mai aproape de realitate.

În cazul în care cel puțin una dintre condiții nu este respectată, atunci obiectul nu poate fi prins. Acest lucru implică lăsarea obiectului din mână dacă vorbim că gestul de prindere este în curs de desfășurare.

```
LM = new Controller();
if (LM.IsConnected)
{
    Debug.Log("Leap Motion conectat");
}
else
{
    Debug.Log("Leap Motion nu este conectat!");
}
```

Figura 4.1.1.2 Verificare conectivitate LM

În scriptul din figura 4.1.1.2 se verifică cu ajutorul proprietății *IsConnected* dacă obiectul *LM* de tipul *Controller* este conectat. Dacă acesta este conectat se printează un mesaj utilizatorului prin care este anunțat de disponibilitatea dispozitivului LM. Acest script este rulat în metoda *Start()* care se execută o singură dată pe scenă în Unity. Verificarea conectivității dispozitivului LM se face la fiecare script dezvoltat pentru fiecare gest construit fiind o condiție esențială pentru ca acesta să poată fi realizat.

Scriptul C# ilustrat în figura 4.1.1.3 ilustrează algoritmul de prindere pentru gestul de prindere 1. În prima buclă *foreach* se face o căutare a mâinilor în fiecare frame, apoi se verifică prim proprietățile *IsRight* și *IsLeft* dacă este cel puțin una dintre mâini prezentă în frame. Dacă este cel puțin o mână prezentă, atunci se trece la parcurgerea tuturor degetelor prezente în frame de la acea mână printr-o altă buclă *foreach*. Aici se verifică prin metoda *Type()* dacă degetele prezente sunt degetul mare și cel arătător. Fiecare dintre acestea are asociat o proprietate cu numele *TYPE_THUMB* și respectiv *TYPE_INDEX.* Aceste proprietăți care fac referire la mână sau degetele de la mână sunt construite special pentru recunoașterea acestora de pachetul software al LM. Punctul unde se găsește degetul mare de la mână este aflat și se inițializează 3 variabile de tip real *thumb_x, thumb_y și thumb_z* cu ajutorul proprietății *TipPosition.x/y/z*. La fel se procedează și pentru degetul arătător. Variabile asociate acestuia sunt: *index_x, index_y și index_z*. Cele trei variabile d1, d2 și d3 semnifică distanțele menționate mai sus, acestea calculând-se prin metoda *distanta().* În ultimul *if* din acces script se verifică valorile distanțelor calculate, iar dacă acea respectă condiția, se afișează un mesaj care anunță utilizatorul că gestul de prindere 1 a avut loc, iar obiectul 3D primește poziția palmei mâinii cu care a fost prins cu ajutorul proprietății *PalmPosition.* Acest script este executat în metoda *Update()* din Unity, care este o corutină. Aceasta are rolul de a executa un script până la o perioadă de timp unde este pusă pe pauză, iar după ce se pornește acesta reia scriptul de la locul unde a fost lăsat. Spre exemplu dacă s-a ajuns la o poziție în spațiul 3D Unity, la reluarea script-ului acesta va continua de la acea poziție nu de la poziția inițială.

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight || m.IsLeft)
    {
        foreach (var f in m.Fingers)
        {
            if (f.Type() == Finger.FingerType.TYPE_THUMB)
            {
                //pozitie deget mare
                thumb_x = f.TipPosition.x;
                thumb_y = f.TipPosition.y;
                thumb_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_INDEX)
            {
                //pozitie deget aratator
                index_x = f.TipPosition.x;
                index_y = f.TipPosition.y;
                index_z = f.TipPosition.z;
            }
            d1 = distanta(thumb_x, thumb_y, thumb_z,index_x,index_y,index_z);
            d2 = distanta(thumb_x, thumb_y, thumb_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            d3 = distanta(index_x,index_y,index_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            if (d1 <= obiect_3D.transform.localScale.x && d2 <= 0.1f && d3 <= 0.1f)
            {
                Debug.Log("Gest prindere 1 - deget mare-aratator");
                obiect_3D.transform.position = new Vector3(m.PalmPosition.x,m.PalmPosition.y,m.PalmPosition.z);
            }
        }
    }
}
```

Figura 4.1.1.3 Script algoritm de detectare al gestului de prindere 1

4.1.2. Gest de prindere 2

Gestul de prindere 2 este definit de degetul arătător de la o singură mână. Implică acțiunea de apropiere și depărtare a degetului arătător față de un obiect. Acesta implică aplicarea unui efect magnetic dintre degetul mare și obiecte dacă distanța dintre acestea este una optimă. În practică, la apropierea degetului utilizatorului de un obiect, obiectul este atras printr-un efect magnetic, lipindu-se de acesta. Utilizatorul arată cu degetul/pointează spre obiectul pe care dorește să-l prindă.

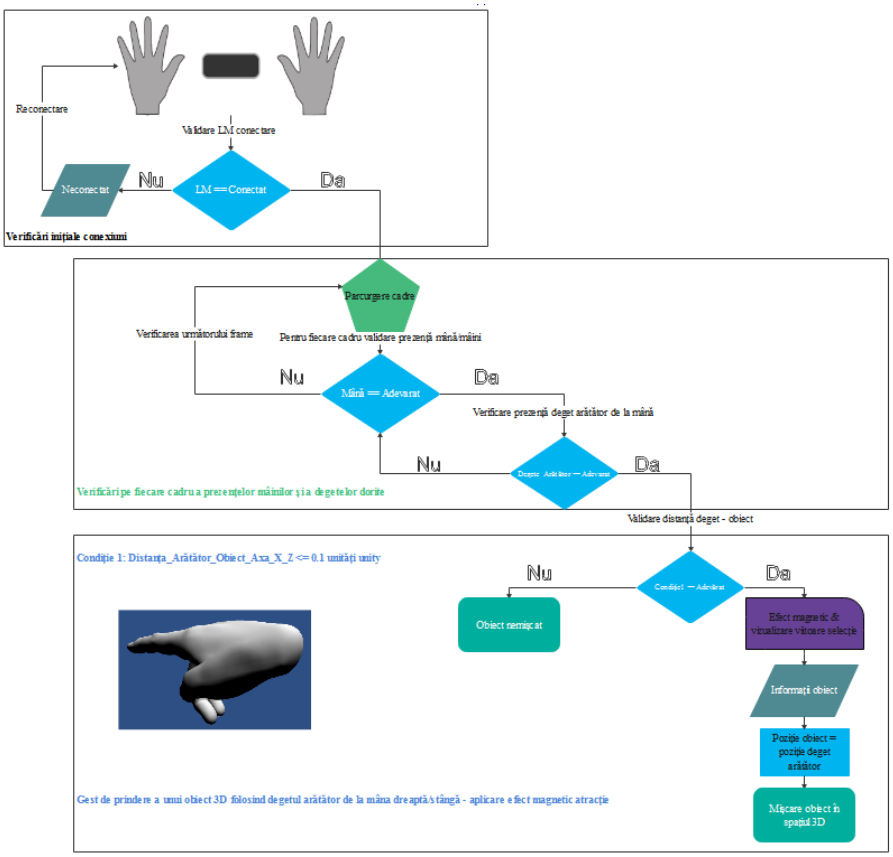

Figura 4.1.2.1

În schema bloc din Figura 4.1.2 sunt exprimați pașii ce trebuie făcuți pentru detectarea gestului definit. La primul pas se face o verificare a prezenței conectivității

dispozitivului LM la PC/laptop. Dacă acesta este conectat se trece la următorul pas, altfel se reia verificarea conectivității. Pasul al doilea implică parcurgerea fiecărui frame pentru identificarea prezenței/absenței mâinilor. Dacă cel puțin o mână este prezentă, atunci se trece la verificarea prezenței degetului arătător. Acest lucru implică ca degetul arătător să fie distins corect și ca acesta să fie extins, nu strâns în pumn. Dacă degetul arătător este prezent atunci se trece la ultimul pas. Aici se verifică distanța degetului arătător față de obiect/obiecte. Acesta implică verificarea următoarei inecuații:

Dist <=*0.1* unități Unity

Unde:

- *Dist*: este distanța dintre degetul arătător și obiect.

Din inecuația descrisă se poate deduce că dacă indexul/degetul arătător este la o distanță aflată în intervalul (0; 0.1] față de obiect, atunci se execută gestul de prindere definit.

Prinderea obiectului implică aplicarea următoarelor funcționalități în scena 3D:

- Informarea utilizatorului despre obiectul prins (ex. Numele obiectului);
- Schimbarea culorii obiectului;
- Atribuirea aceleiași valori pentru poziția obiectului la fel cu poziția degetului arătător;
- Mișcarea obiectului prin spațiul 3D.

Dacă obiectul este lipit de degetul arătător poate fii mișcat prin spațiul 3D. Lăsarea acestuia implică folosirea efectului de coliziune cu un alt obiect. Practic când obiectul prins se atinge cu altul în scenă se poate lăsa obiectul. Un alt mod de a lăsa obiectul este de a verifica dacă acesta se găsește pe o arie definită de utilizator.

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight || m.IsLeft)
    {
        foreach (var f in m.Fingers)
        {

            if (f.Type() == Finger.FingerType.TYPE_INDEX)
            {
                //pozitie deget aratator
                index_x = f.TipPosition.x;
                index_y = f.TipPosition.y;
                index_z = f.TipPosition.z;
            }
            d = distanta(index_x, index_y, index_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            if (d <= 0.1f)
            {
                Debug.Log("Gest prindere 2 - deget aratator");
                obiect_3D.transform.position = new Vector3(index_x, index_y, index_z);
            }
        }
    }
}
```

Figura 4.1.2.2 Script algoritm de detectare al gestului de prindere 2

Scriptul din figura 4.1.2.2 ilustrează cum este construit și detectat gestul de prindere 2. Acesta cuprinde o buclă *foreach* prin care se parcurg frameurile în scopul detectării mâinilor, apoi prin proprietățile *IsRight* și *IsLeft* se verifică ce mână este detectată. Următoarea buclă *foreach* parcurge degetele de la mână pentru a detecta degetul arătător prin proprietatea *TYPE_INDEX*. Dacă degetul arătător este prezent, atunci se inițializează variabilele reale *index_x, index_y și index_z* cu poziția degetului în spațiul virtual 3D. O dată cu inițializarea variabilelor se calculează distanța dintre degetul mare și obiectul 3D cu ajutorul metodei distanță ce returnează distanța dintre două puncte. Ultima instrucțiune decizională *if* verifică valoarea distanței calculate. Dacă aceasta este mai mică decât 0.1, atunci utilizatorul este anunțat printr-un mesaj ca gestul de prindere 2 a fost executat și obiectul ce trebuie prins se va poziționa la vârful degetului arătător cu ajutorul proprietății *position.*

4.1.3. Gest de prindere 3

În figura 4.1.3 este prezentată schema bloc a algoritmului de detecție a gestului de prindere comlplex [3NIC18]. La fel ca în ceilalți doi algoritmi pentru gestul de prindere la primul pas se verifică conectivitatea dispozitivului LM. Dacă acesta este conectat, se trece la pasul al doilea. Aici se parcurge fiecare frame și se detectează prezența uneia sau a ambelor mâini. Dacă cel puțin o mână este prezentă se verifică dacă sunt prezente degetul mare și cel puțin încă un deget. Apoi se verifică trei distanțe: distanța dintre degetul mare și oricare alt deget/degete (Dist_1) prezent în acel frame, distanța dintre degetul mare și obiect (Dist_2) și distanța dintre oricare alt deget/degete prezente în frame și obiect (Dist_3). Fiind vorba de un gest dinamic se vor forma trei intervale de valori pentru cele trei distanțe luate în considerare. Astfel vom avea trei inecuații:

Dist_1 <= dimensiune obiect

Dist_2 <= 0.1 unități Unity

Dist_3 <= 0.1 unități Unity

Dacă cele trei condiții sunt respectate, atunci gestul de prindere este efectuat, iar acesta are ca efect lipirea obiectului ce trebuie prins de palma mâinii. Astfel poziția obiectului va fi egală cu poziția palmei atât timp cât cele trei condiții sunt respectate.

Prinderea obiectului vine împreună cu alte funcționalități/efecte 3D care au ca scop îmbunătățirea interactivității dintre om și calculator. Printre funcționalități/efecte 3D amintim: informarea utilizatorului despre numele obiectului prins, colorarea obiectului/schimbarea culorii obiectului prins (funcționalitate de selecție), mișcarea obiectului în spațiul 3D.

Când una dintre cele trei condiții nu mai este respectata, atunci obiectul este lăsat din mână și atunci are loc terminarea gestului de prindere.

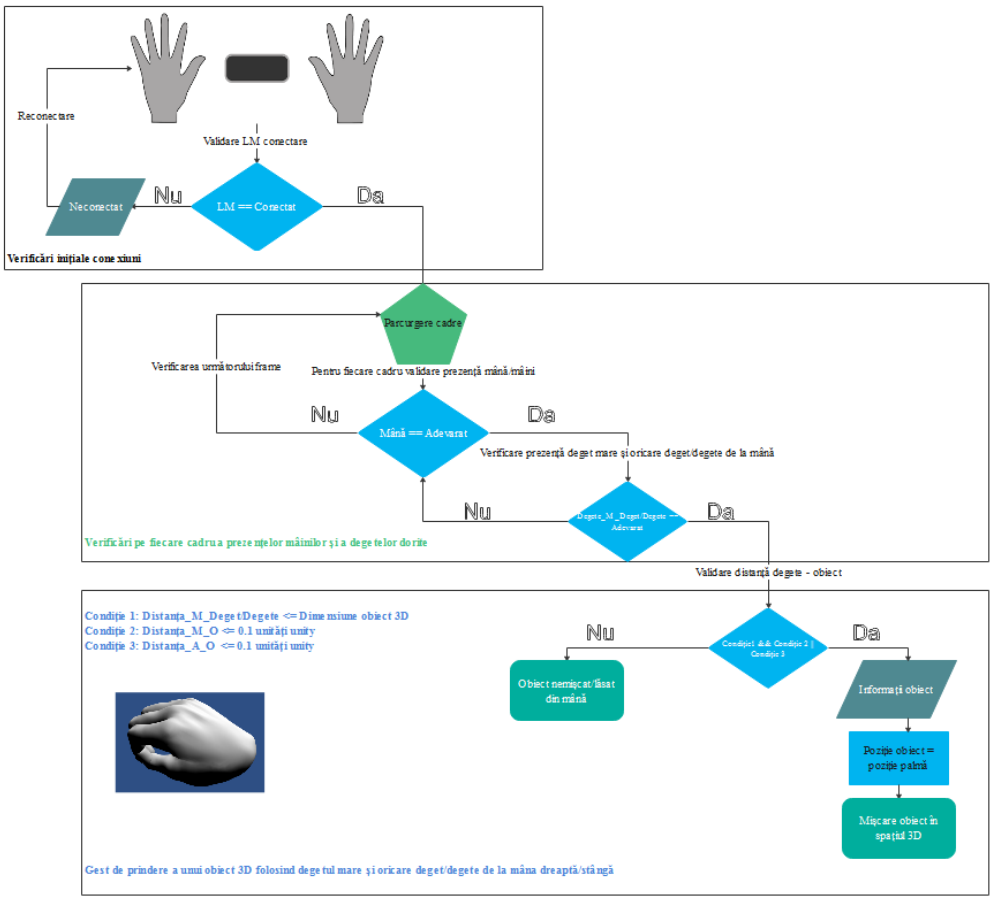

Figura 4.1.3.1

Scriptul din figura 4.1.3.1. ilustrează principalele metode și proprietăți folosite pentru construirea și detectarea gestului de prindere 3. La fel ca și la celelalte scripturi descrise la precedentele gesturi pentru început se parcurg frameurile în scopul detectării mâinilor cu ajutorul proprietăților *IsRight și IsLeft.* Dacă este cel puțin o mână prezentă atunci se parcurge printr-o buclă *foreach* fiecare frame cu scopul de a detecta degetele de la mână. Tipul degetului detectat în frame se face cu una dintre proprietățile: *TYPE_THUMB* pentru degetul mare, *TYPE_INDEX* pentru degetul arătător, *TYPE_MIDDLE* pentru degetul mijlociu, *TYPE_RING* pentru degetul inelar și *TYPE_PYNKY* pentru degetul mic de la mână. O dată găsit tipul degetului, se fac 15 inițializări de variabile. Grupuri de câte 3 variabile inițializate cu poziția fiecărui deget în spațiul 3D. Apoi se calculează 9 distanțe cu ajutorul metodei *distanta().* Aceste 9 distanțe fac referire la:

- Distanța dintre degetul mare și degetul arătător;
- Distanța dintre degetul mare și degetul mijlociu;

- Distanța dintre degetul mare și degetul inelar;
- Distanța dintre degetul mare și degetul mic;
- Distanța dintre degetul mare și obiectul 3D;
- Distanța dintre degetul arătător și obiectul 3D;
- Distanța dintre degetul mijlociu și obiectul 3D;
- Distanța dintre degetul inelar și obiectul 3D;
- Distanța dintre degetul mic și obiect;

Cele 9 distanțe sunt calculate cu scopul de a acoperii toate combinațiile de degete pe care putem să le folosim pentru a prinde un obiect.

După calcularea distanțelor se verifică dacă oricare dintre primele 4 distanțe sunt mai mici decât suprafața obiectului ce trebuie prins (scala obiectului pe axa x) cu ajutorul proprietății *localScale*. Ultimele verificări din aceiași instrucțiune *if* verifică dacă cel puțin una dintre ultimele 5 distanțe menționate mai sus sunt mai mici decât 0.1 unități Unity.

Respectarea condițiilor menționate duc la formarea gestului de prindere 3, utilizatorul este înștiințat că acesta a avut loc, iar obiectul ce dorește să-l prindă se lipește de palmă primind poziția palmei cu ajutorul proprietății *PalmPosition.*

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight || m.IsLeft)
    {
        foreach (var f in m.Fingers)
        {
            if (f.Type() == Finger.FingerType.TYPE_THUMB)
            {
                //pozitie deget mare
                thumb_x = f.TipPosition.x;
                thumb_y = f.TipPosition.y;
                thumb_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_INDEX)
            {
                //pozitie deget aratator
                index_x = f.TipPosition.x;
                index_y = f.TipPosition.y;
                index_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_MIDDLE)
            {
                //pozitie deget mijlociu
                middle_x = f.TipPosition.x;
                middle_y = f.TipPosition.y;
                middle_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_RING)
            {
                //pozitie deget inelar
                ring_x = f.TipPosition.x;
                ring_y = f.TipPosition.y;
                ring_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_PINKY)
            {
                //pozitie deget mic
                pinky_x = f.TipPosition.x;
                pinky_y = f.TipPosition.y;
                pinky_z = f.TipPosition.z;
            }
            d1 = distanta(thumb_x, thumb_y, thumb_z, index_x, index_y, index_z);
            d2 = distanta(thumb_x, thumb_y, thumb_z, middle_x, middle_y, middle_z);
            d3 = distanta(thumb_x, thumb_y, thumb_z, ring_x, ring_y, ring_z);
            d4 = distanta(thumb_x, thumb_y, thumb_z, pinky_x, pinky_y, pinky_z);
            d5 = distanta(thumb_x, thumb_y, thumb_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            d6 = distanta(index_x, index_y, index_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            d7 = distanta(middle_x, middle_y, middle_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            d8 = distanta(ring_x, ring_y, ring_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
            d9 = distanta(pinky_x, pinky_y, pinky_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);

            if ((d1 <= obiect_3D.transform.localScale.x || d2 <= obiect_3D.transform.localScale.x || d3 <= obiect_3D.transform.localScale.x || d4 <= obiect_3D.transform.localScale.x)
                && (d5 <= 0.1f) && (d6 <= 0.1f || d7 <= 0.1f || d8 <= 0.1f || d9 <= 0.1f))
            {
                Debug.Log("Gest prindere 3 - deget mare-oricare alt deget/degete");
                obiect_3D.transform.position = new Vector3(m.PalmPosition.x, m.PalmPosition.y, m.PalmPosition.z);
            }
        }
    }
}
```

Figura 4.1.3.2 Script algoritm de detectare al gestului de prindere 3

4.2. Algoritmul de detecție a gestului de flexie și extensie a încheieturii mâinii

Acest gest face parte din gesturile construite care au aplicabilitate în recuperarea medicală. Gestul de flexie și extensie a mâinii presupune o mișcare de glisare a palmei mâinii în sus și în jos până la o limită maximă – pentru flexie și minimă pentru extensie.

Algoritmul de detectare a gestului se bazează pe verificarea unghiului de rotație a palmei față de un vector ce este perpendicular pe axa OX. Acesta presupune verificarea conectivității dispozitivului LM, apoi verificarea prezenței uneia dintre mâini în frame-uri, iar ca ultim pas verificarea unghiului format dintre palmă și vectorul perpendicular pe axa OX.

Efectul pe care acest gest îl are în spațiul 3D este mișcarea pe axa OY a unui obiect. Practic utilizatorul poate modifica înălțimea la care se află un obiect 3D în spațiu.

Deoarece acest gest este utilizat în recuperarea mâinilor, acesta a fost împărțit în 3 etape sau nivele de dificultate. Fiecare etapă presupune mișcarea palmei la un unghi de 60 de grade față de axa OY. În continuare, vor fi definite intervalele unghiurilor care definesc etapele/nivelele gestului de flexie și extensie:

Etapa 1: (0; 60]

Etapa 2: (60; 120]

Etapa 3: (120; 180]

Atingerea pragurilor maxime pentru fiecare nivel/etapă constituie începutul unui nou nivel, astfel utilizatorul trece de una dintre fazele recuperării încheieturii mâinii. Prin împărțirea gestului de flexie și extensie în 3 etape, din 60 în 60 de grade se poate face o monitorizare mult mai ușoară asupra nivelului de recuperare al mâinii. Așa cum reiese din acestea, utilizatorul trebuie să facă mișcări mult mai ample/lungi ca să ajungă la ultima etapă de recuperare a mâinii sau a încheieturii acesteia.

Scriptul din figura 4.2.1 arată cum este construit și detectat gestul de flexie și extensie. Acesta presupune parcurgerea printr-o buclă *foreach* a frameurilor pentru a detecta prezența mâinilor. Folosim proprietățile *IsRight* și *IsLeft* pentru a verifica dacă variabila *m* de tipul *Hands* este detectată ca fiind mâna dreaptă sau stângă. Variabila cu numele *gratia* se va inițializa cu valoarea modulului unghiului de inclinare a mâinii cu ajutorul proprietății *Yaw.* Tot la această inițializare se face transformarea unghiului de înclinare din radiani în grade pentru o identificare mai bună a unghiului de înclinare a mâinii. După calcularea unghiului de înclinare/grația mâinii se identifică în ce interval face parte unghiul format. Dacă unghiul nu este mai mare decât 60 de grade atunci se regăsește nivelul 1 al gestului de flexie si extensie al mâinii, dacă este între 60 și 120 de grade este nivelul 2 al gestului de flexie și extensie al mâinii, iar dacă este mai mare decât 120 este nivelul 3. Se poate observa ca în acest script este important găsirea mâinii în frame și detectarea înclinării mâinii cu ajutorul proprietății *Yaw*.

Acest script este utilizat în interiorul metodei *Update()* ce se execută o dată pe frame în Unity.

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight || m.IsLeft)
    {
        Vector dir = m.Direction;
        var gratia = Math.Abs(dir.Yaw * (180.0f / Math.PI));

        if (gratia > 0 && gratia <= 60)
            Debug.Log("Gest flexie_extensie - nivel 1");
        else if (gratia > 60 && gratia <= 120)
            Debug.Log("Gest flexie_extensie - nivel 2");
        else
            Debug.Log("Gest flexie_extensie - nivel 2");
    }
}
```

Figura 4.2.1 Script algoritm de detectare al gestului de flexie și extensie

4.3. Algoritmul de detecție a gestului de rotire al palmei

Gestul de rotire al palmei se numește gestul de pronație și supinație. Când palma mâinii se află către utilizator mișcarea este de supinație, iar când aceasta este ascunsă este pronație. Mișcarea de rotire pentru descoperirea și ascunderea palmei constituie gestul de rotire al palmei. Acest gest dinamic are aplicabilitate în recuperarea mâinilor, antrenând încheietura mâinii.

Pentru detectarea gestului de către dispozitivul LM, se verifică conectivitatea acestuia, iar apoi prezența mâinii în fiecare cadru realizat de camerele dispozitivului LM. Dacă este prezentă una dintre mâinii, se verifică dacă unghiul format dintre palmă și axa OX este mai mic decât 45 de grade, atunci este contorizat ca începutul gestului de pronație. Formarea unui unghi dintre palmă și axa OX mai mare decât 90 de grade implică începutul gestului de supinație. Executarea unui gest complet de rotire al palmei implică executarea unei rotații de 180 de grade a palmei față de axa OX. Astfel se ating ambele gesturi, de pronație și supinație.

Pentru o aplicabilitate mai bună a acestui gest în recuperarea mobilității mâinilor, acesta a fost împărțit în patru nivele. Fiecare nivel având un prag pe care utilizatorul trebuie să-l atingă în rotirea mâinii. Rotirea mâinii pe un unghi mai mare solicită încheietura mâinii tot mai tare, astfel împărțirea gestului în 4 etape este folositoare când ne referim la recuperarea încheieturii mâinii. Pentru a avea o monitorizare cât mai simplă asupra nivelului de recuperare a mâinii s-a creat intervale ale gestului de rotire din 45 în 45 de grade.

În continuare vor fi enumerate nivelurile cu intervalul de valori pentru unghiul de rotire al palmei:

Nivel_1 = (0; 45]

Nivel_2 = (45; 90]

Nivel_3 = (90; 135]

Nivel_4 = (135; 180]

Primele două nivele implică gestul de pronație, iar următoarele două implică gestul de supinație.

Cu cât utilizatorul poate să rotească mai mult încheietura mâinii cu atât acesta poate ajunge la o recuperare completă a acesteia.

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight || m.IsLeft)
    {
        Vector normal = m.PalmNormal;
        var rotire = Math.Abs(normal.Roll * (180.0f / Math.PI));
        if (rotire > 0 && rotire <= 45)
            Debug.Log("Gest rotire (pronatie_supinatie) - nivel 1");
        else if (rotire > 45 && rotire <= 90)
            Debug.Log("Gest rotire (pronatie_supinatie) - nivel 2");
        else if (rotire > 90 && rotire < 135)
            Debug.Log("Gest rotire (pronatie_supinatie) - nivel 3");
        else
            Debug.Log("Gest rotire (pronatie_supinatie) - nivel 4");

    }
}
```

Figura 4.3.1 Script algoritm de detectare al gestului de flexie și extensie

În scriptul din figura 4.3.1 se exemplifică cum este detectat gestul de rotire al mâinii ce semnifică pronația și supinația mâinii. La fel ca în celelalte gesturi se pleacă de la parcurgerea printr-o buclă *foreach* a frameurilor pentru a detecta prezența mâinilor în cadru cu ajutorul proprietăților *IsRight* și *IsLeft*. Dacă este prezentă o mână în cadru, atunci se inițializează variabila *rotire* cu valoarea modulului unghiului de rotație al mâinii. Valoarea unghiului de rotație este dată de proprietatea *Roll* care returnează unghiul de rotație exprimat în radiani. Astfel pentru o vizualizare și împărțire mai bună a intervalelor unghiului de rotație s-a transformat valoarea acestuia din radiani în grade. Prin blocurile de instrucțiuni *if-else* se face împărțirea gestului în cele 4 nivele. Astfel dacă valoarea unghiului este mai mică sau egală decât 45 de grade este identificat nivelul 1, dacă valoarea unghiului este mai mare decât 45 și mai mică decât 90 de grade este nivelul 2, dacă valoarea este între 90 și 135 este nivelul 3 și dacă este mai mare de 135 este nivelul 4.

La fel ca si celelalte scripturi ce definesc algoritmii de detecție a gesturilor create și acest script este executate în metoda *Update()* din Unity.

4.4. Algoritmul de detecție a gestului de strângere și deschidere al mâinii

Tot un gest cu aplicabilitate în recuperarea mâinilor este gestul de strângere și deschidere a mâinii. Acesta pune în exercițiu articulațiile degetelor, practic se exemplifică o flexie și o extensie a degetelor.

Algoritmul de detectare al acestui gest seamănă cu algoritmul de detectare a gestului de prindere al unui obiect 3D. Primii doi pași din algoritmul de prindere al unui obiect sunt respectați și aici: verificarea conectivității dispozitivului LM (pasul 1) și verificarea prezenței uneia dintre mâini în cadru (pasul 2). La următorul pas, pentru a verifica cât de mult este deschisă/închisă mâna s-a introdus calcularea semiperimetrului format de figura geometrică descrisă de următoarele puncte: coordonatele degetului mare, a degetului mijlociu și podul palmei. După cum se observă din aceste coordonate, figura geometrică care se formează este un triunghi. Deoarece acest gest are aplicabilitate în recuperarea articulațiilor degetelor mâinilor, gestul a fost împărțit în trei nivele, fiecare nivel fiind descris de un interval de valori al semiperimetrului figurii geometrice descrise. Astfel dacă semiperimetrul *sP* este mai mare decât 150 de unități în spațiul Unity, atunci mâna este complet deschisă (extensia degetelor este completă), dacă semiperimetrul *sP* este cuprins în intervalul [150; 100] mâna este închisă la jumate sau întredeschisă, iar dacă semiperimetrul *sP* are o valoare mai mică decât 100 mâna este complet închisă, sau pumnul este strâns (flexia degetelor este completă).

Aceste intervale de valori ale semiperimetrului sunt alese pentru o scală a mâinii virtuale folosită egală cu 25 de unități Unity. Solicitarea tot mai mare a articulațiilor degetelor se face la strângerea pumnului. Deoarece vorbim despre recuperarea articulațiilor degetelor de la mână, am ales împărțirea în intervale de valori pentru semiperimetrul generat.

sP > 150 unități Unity

sP = [150; 100] unități Unity

sP < 100 unități Unity

Scriptul din figura 4.4.1 presupune executarea celorlalți doi pași din algoritmul de detectare a gestului de deschidere și închidere a mâinii.

Pentru început prin prima buclă repetitivă *foreach* se caută în fiecare *frame* mâinile, dacă se găsește o mână și aceasta este dreapta, se inițializează poziția palmei. Pentru a verifica dacă mâna este dreapta sau stânga, proprietatea de tipul *LM* cu numele *IsRight* este prezentă, respectiv *IsLeft.* Următoarea buclă repetitivă, care se găsește în interiorul primeia, verifică dacă se pot distinge degete la mâna găsită în frame. Dacă degete sunt prezente în frame, atunci se verifică în cele două instrucțiuni decizionale dacă tipul degetului este mare și mijlociu. Acest lucru este realizat tot cu ajutorul proprietăților de tipul *LM* cu numele *TYPE_THUMB* și *TYPE_MIDDLE*. O dată

inițializate pozițiile degetelor mare și mijlociu, se poate calcula distanțele dintre acestea și dintre ele și poziția palmei. Distanța se calculează cu ajutorul metodei *distanță()* care va returna distanța dintre două puncte în spațiul 3D. În final se calculează semi perimetrul figurii geometrice create cu ajutorul celor trei puncte și a distanțelor dintre ele. Metoda *semiPerimetru()* realizează acest lucru. Ultimele trei instrucțiuni decizionale anunță utilizatorul despre nivelul gestului de închidere și deschidere a mâinii în funcție de valoarea semi perimetrului calculat.

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight)
    {
        //pozitie palma
        palm_x = m.PalmPosition.x;
        palm_y = m.PalmPosition.y;
        palm_z = m.PalmPosition.z;
        foreach (var f in m.Fingers)
        {
            if (f.Type() == Finger.FingerType.TYPE_THUMB)
            {
                //pozitie deget mare
                thumb_x = f.TipPosition.x;
                thumb_y = f.TipPosition.y;
                thumb_z = f.TipPosition.z;
            }
            if (f.Type() == Finger.FingerType.TYPE_MIDDLE)
            {
                //pozitie deget mijlociu
                middle_x = f.TipPosition.x;
                middle_y = f.TipPosition.y;
                middle_z = f.TipPosition.z;
            }
            d1 = distanta(palm_x, palm_y, palm_z, thumb_x, thumb_y, thumb_z);
            d2 = distanta(palm_x, palm_y, palm_z, middle_x, middle_y, middle_z);
            d3 = distanta(thumb_x, thumb_y, thumb_z, middle_x, middle_y, middle_z);
            sP = semiPerimetru(d1, d2, d3);
            if (sP > 150)
                Debug.Log("Gest deschidere - inchidere mana nivel 1");
            if (sP >= 150 && sP <= 100)
                Debug.Log("Gest deschidere - inchidere mana nivel 2");
            if (sP < 100)
                Debug.Log("Gest deschidere - inchidere mana nivel 3");
        }
    }
}
```

Figura 4.4.1 Script algoritm de detectare gest închidere deschidere mână

4.5. Algoritmul de detecție a gestului complex

Dacă de cele mai multe ori, când ne referim la un gest construit pentru interacțiunea dintre om și calculator, facem apel la folosirea unei singure mâini, la gestul complex este vorba de folosirea ambelor mâini. Pentru executarea acestui gest utilizatorul apropie sau depărtează mâinile una față de alta. Gestul complex construit

este un gest dinamic care începe la prezența mâinilor în cadru și a unui obiect între acestea, și se termină la absența a cel puțin unei mâini din cadru.

Algoritmul de detectare a gestuli complex presupune efectuarea a trei pași. La fel ca și la celelalte gesturi în primul pas se face verificarea conectivității dispozitivului LM. Dacă acesta este conectat, se verifică prezența mâinilor pentru fiecare frame (pasul 2). La pasul al treilea se verifică distanța dintre mâini și prezența unui obiect între acestea. Dacă obiectul este prezent între acestea, atunci în funcție de distanța dintre mâini, obiectul se mărește sau se micșorează. O distanță care crește între mâini implică mărirea obiectului, iar o distanță care se micșorează implică micșorarea obiectului.

Oprirea gestului complex de mărire/micșorare este dată de dispariția unei mâini din frame (în cazul măririi obiectului), sau atingerea unui prag de valori minim setat la 0.1 unități Unity în spațiul 3D pentru distanța dintre mâini (în cazul micșorării obiectului).

```
frame = LM.Frame();
foreach (var m in m_c.GetFrame().Hands)
{
    if (m.IsRight)
    {
        //pozitie palma mana dreapta
        palm_r_x = m.PalmPosition.x;
        palm_r_y = m.PalmPosition.y;
        palm_r_z = m.PalmPosition.z;
    }
    if (m.IsLeft)
    {
        //pozitie palma mana dreapta
        palm_l_x = m.PalmPosition.x;
        palm_l_y = m.PalmPosition.y;
        palm_l_z = m.PalmPosition.z;

    }
    d1 = distanta(palm_r_x, palm_r_y, palm_r_z, palm_l_x, palm_l_y, palm_l_z);
    d2 = distanta(palm_r_x, palm_r_y, palm_r_z, obiect_3D.transform.position.x,obiect_3D.transform.position.y,obiect_3D.transform.position.z);
    d3 = distanta(palm_l_x, palm_l_y, palm_l_z, obiect_3D.transform.position.x, obiect_3D.transform.position.y, obiect_3D.transform.position.z);
    if (d1 > 3f && d2 > obiect_3D_initial.transform.localScale.x && d3 > obiect_3D_initial.transform.localScale.x && obiect_3D.transform.localScale.x > 100f)
    {
        Debug.Log("Gest complex - marire obiect");
        obiect_3D.transform.localScale += new Vector3(0.1f, 0.1f, 0.1f);
    }
    if(d1 <= 3f && d2 <= obiect_3D_initial.transform.localScale.x && d3 <= obiect_3D_initial.transform.localScale.x && obiect_3D.transform.localScale.x > 1f)
    {
        Debug.Log("Gest complex - micsorare obiect");
        obiect_3D.transform.localScale -= new Vector3(0.1f, 0.1f, 0.1f);
    }
}
```

Figura 4.5.1 Script algoritm de detectare a gestului complex

În scriptul din figura 4.5.1 este exemplificat cum este creat și detectat gestul complex. Pentru început se parcurge fiecare frame din LM printr-o buclă *foreach*, apoi sunt detectate prezențele celor două mâini cu ajutorul proprietăților *IsRight* și *IsLeft*. Dacă acestea sunt prezente se inițializează 6 variabile: *palm_r_x, palm_r_y, palm_r_z, palm_l_x, palm_l_y, palm_l_z* cu poziția celor două palme în spațiul 3D pentru toate cele 3 axele. Se calculează distanța dintre cele două palme ale mâinilor prin metoda distanta() și distanțele dintre palme și obiectul 3D tot prin aceiași metodă.

Cel două instrucțiuni *if* de la sfârșit verifică valorile celor 3 distanțe calculate. Dacă distanța dintre cele două palme este mai mare decât 3 unități Unity și distanțele dintre palme și obiect sunt mai mari decât scala inițială a obiectului 3D pe axa x și scala/mărimea obiectul 3D pe axa x este mai mică decât 100 de unități Unity, atunci

are loc gestul complex, iar obiectul 3D se mărește cu 0.1 unități pe fiecare axă. Micșorarea obiectului tot cu aceeași valoare ca și mărirea are loc când distanța dintre palme este mai mică sau egală decât 3 unități Unity, distanțele dintre palme și obiect sunt mai mici sau egale decât scala obiectului 3D inițial pe axa x și obiectul 3D este mai mare decât o unitate Unity pe scala x.

4.6. Concluzii

În acest capitol s-a prezentat algoritmii creați pentru detectarea gesturilor dinamice definite. Gesturile create sunt: gestul de prindere 1, 2 și 3, gestul de flexie și extensie, gestul de rotire a mâinii – pronație și supinație, gestul de strângere și deschidere a mâinii și gestul complex de apropiere și depărtare al mâinilor una față de cealaltă.

Contribuțiile aduse în domeniul Calculatoarelor și tehnologiei informației prezentate în acest capitol sunt reprezentate de algoritmii creți cu scop în detectarea celor mai folosite gesturi în controlul unor interfețe 3D și a celor mai utilizate gesturi în recuperarea mișcării și a articulațiilor mâinilor.

Algoritmii creați se bazează pe parcurgere de frameuri Leap Motion în vederea detecției poziției mâinilor în spațiul virtual 3D și pe calcularea prin formule matematice a distanțelor dintre două puncte, a unghiurilor formate și a suprafețelor și/sau semiperimetrelor formate de punctele detectate.

Pentru a ajuta utilizatorii în controlul interfețelor 3D au fost abordate mai multe moduri de a detecta un gest care presupune prinderea unui obiect 3D. S-au descriși 3 algoritmi pentru detectarea gestului de prindere al obiectelor.

Totodată au fost descrise și 3 gesturi care au aplicabilitate în recuperare, acestea stând la baza exercițiilor pe care utilizatorii trebuie să le facă pentru recuperarea mișcării mâinilor.

Întrucât în literatura de specialitate sunt prezente puține abordări de creare de gesturi ce implică folosirea ambelor mâini pentru dispozitivul Leap Motion, s-a creat un algoritm pentru detectarea unui gest complex. Gestul creat are aplicabilitate în manipularea obiectelor 3D cu scopul de a mării sau micșora scala acestora.

Am propus 7 algoritmi de detecție a gesturilor. Principala provocare care urmează după definirea acestora este clasificarea cu o acuratețe foarte ridicată a gesturilor pentru dispozitivul Leap Motion. Eliminarea confuziilor dintre gesturi și creșterea preciziei de detecție a acestora este următoarea provocare care va fi discutată în capitolul următor.

5. MODELE DE CLASIFICARE CARE CONTRIBUIE LA CREȘTEREA PRECIZIEI GESTURILOR DINAMICE

Capitolul acesta este destinat prezentării contribuțiilor avute la precizia și acuratețea gesturilor dinamice create pentru dispozitivul Leap Motion (LM). Criteriul de clasificare al gesturilor LM, este legat de folosirea rețelelor neuronale convoluționale [NIC19]. Algoritmii folosiți pentru clasificarea gesturilor contribuie la o precizie și acuratețe ridicată.

Motivația care stă la baza acestui capitol este: eliminarea confuziilor și detectărilor eronate ale gesturilor

LM definite în capitolul 4.

Exemplu:

- Gestul de rotire al mâinii - mișcarea de pronație și supinație este detectat greșit datorită orientării degetului mare de la mână, care duce la o confuzie în ceea ce privește folosirea mâinii virtuale corecte.

Diversitatea datelor colectate prin intermediul dispozitivului LM, contribuie la precizia și acuratețea gesturilor, a căror algoritmi au fost explicați în capitolul precedent.

În continuare vor fi explicați principalii pași pe care i-am urmat pentru a crește precizia gesturilor dinamice create:

- Colectarea datelor de la dispozitivul LM
- Filtrarea datelor colectate/curățarea datelor
- Selectarea modelului rețelei neuronale folosit și clasificarea datelor
- Prezentarea metricilor pentru rețeaua neuronală folosită.
- Evaluarea rezultatelor obținute

5.1. Colectarea datelor de la dispozitivul LM

Gesturile pentru care s-au colectat date sunt gesturile a căror aplicabilitate o reprezintă recuperarea mișcării mâinilor. Astfel s-au realizat algoritmii de detecție a gesturilor folosite în recuperare explicați în capitolul 4. În continuare se vor prezenta tipurile de date colectate pentru fiecare gest de recuperare:

- gestului de strângere și deschidere al mâinii (gestul 1)
- gestului de rotire al palmei (gestul 2)
- gestului de flexie și extensie al mâinii (gestul 3)

Pentru primele două tipuri de gesturi prezentate mai sus s-a realizat același set de date, care dispune de o diversitate mare de valori. Un total de 29 de caracteristici au fost colectate. Acest număr mare de date ce descriu cele două gesturi au fost selectate pentru a crește acuratețea și precizia de clasificare a gesturilor.

În continuare vor fi prezentate cel 29 tipuri de date:

- 15 date distincte ce descriu poziția fiecărui deget de la mână pe fiecare axă x, y, z, din spațiul 3D.
- 5 distanțe dintre degete și palmă (distanța dintre vârful degetelor și punctul din mijlocul palmei).
- 3 valori pentru fiecare axă a vectorului direcție al degetului mare.
- 4 distanțe dintre degetul mare și vârful celorlalte degete.

- Unghiul de rotire al mâinii față de un punct fix în coordonatele (0, 0, 0).
- Clasa din care face parte gestul. Valori cuprinse în intervalul de numere naturale [1, 6].

Intervalele de valori la care se face referire pentru datele de mai sus sunt descrise în capitolul 4. Vectorul direcție cât și unghiul de rotire introdus în categoriile de tipuri de date colectate, au fost alese pentru a crește precizia de clasificare a gestului de rotire al palmei. De multe ori s-a observat că la folosirea acestui gest se produceau confuzii legate de ce mână este folosită, dreapta sau stânga. Distanțele precum și pozițiile degetelor introduse în descrierea gesturilor au fost alese pentru a crește precizia de clasificare a gestului de strângere și deschidere al mâinii.

Datele au fost colectate într-un fișier text a căror valori arată ca și în tabelele 5.1 și 5.2

Poziție deget mare axa X	-76.39787	-74.73158	-57.84035
Poziție deget mare axa Y	150.2195	184.5612	200.4308
Poziție deget mare axa Z	9.287013	-65.9808	-64.98376
Poziție deget arătător axa X	-37.90823	-15.48665	-10.69128
Poziție deget arătător axa Y	160.1073	163.93.48	182.494
Poziție deget arătător axa Z	-97.17207	-56.29712	-50.78408
Poziție deget mijlociu axa X	-9.510086	-5.774602	-3.328459
Poziție deget mijlociu axa Y	172.5841	159.6167	176.297
Poziție deget mijlociu axa Z	-116.1482	-52.61379	-47.82751
Poziție deget inelar axa X	26.05887	0.078151128	10.30481
Poziție deget inelar axa Y	169.2073	150.7754	179.249
Poziție deget inelar axa Z	-114.0639	-41.38922	-37.88974
Poziție deget mic axa X	65.25019	14.50092	22.69773
Poziție deget mic axa Y	162.7072	162.2424	190.2383
Poziție deget mic axa Z	-93.29159	-39.22274	-35.41889

Distanță deget mare - arătător	113.63	63.48	52.41
Distanță deget mare - mijlociu	143.9	74.54	62.03
Distanță deget mare - inelar	161.47	85.69	76.33
Distanță deget mare - mic	175.34	95.79	86.4
Distanță deget mare - palmă	95.81	84.37	78.43
Distanță deget arătător - palmă	102.26	57.11	55.38
Distanță deget mijlociu - palmă	109.31	58.08	58.1
Distanță deget inelar - palmă	104.11	62.45	52.05
Distanță deget mic - palmă	95.77	54.45	45.65
Vector direcție deget mare axa X	-0.452630930668918	-0.356264282389353	-0.264719319042548
Vector direcție deget mare axa Y	0.88999875463208	0.879849915784268	0.917316384614892
Vector direcție deget mare axa Z	0.0550223350571013	-0.314547113781092	-0.297412731137578
Unghiul de rotire al mâinii	72.150520324707	60.7153549194336	65.4316558837891
Clasă gest	1	2	3

Tabelul 5.1 Exemplu valori colectate pentru gestul de strângere și deschidere al mâinii

Poziție deget mare axa X	-72.7239	1.80064	94.2042
Poziție deget mare axa Y	113.1621	273.92	242.2868
Poziție deget mare axa Z	6.066475	-17.37169	-17.95954
Poziție deget arătător axa X	-29.8458	-29.55146	15.81545
Poziție deget arătător axa Y	141.0819	198.313	205.8427
Poziție deget arătător axa Z	-105.442	-129.0831	-126.5771
Poziție deget mijlociu axa X	-6.02518	-24.20794	-15.53861

Poziție deget mijlociu axa Y	165.1587	161.5371	171.9714
Poziție deget mijlociu axa Z	-120.539	-140.8126	-136.1337
Poziție deget inelar axa X	28.53047	-6.873411	-38.73087
Poziție deget inelar axa Y	177.1444	121.7954	145.0626
Poziție deget inelar axa Z	-114.485	-129.5314	-117.9451
Poziție deget mic axa X	66.63553	10.73156	-62.54891
Poziție deget mic axa Y	180.4688	90.97326	125.8353
Poziție deget mic axa Z	-90.7288	-94.88651	-75.56827
Distanță deget mare - arătător	122.69	138.49	138.82
Distanță deget mare - mijlociu	152.25	168.95	175.93
Distanță deget mare - inelar	169.94	189.2	192.67
Distanță deget mare - mic	182.54	198.89	203.6
Distanță deget mare - palmă	98.68	107.06	107.12
Distanță deget arătător - palmă	102.43	104.46	104.35
Distanță deget mijlociu - palmă	109.44	110.34	113.07
Distanță deget inelar - palmă	104.29	107.31	108.97
Distanță deget mic - palmă	96.19	99.87	101.64
Vector direcție deget mare axa X	-0.54009	0.006560279	0.361522844
Vector direcție deget mare axa Y	0.840403	0.997973596	0.92981231
Vector direcție deget mare axa Z	0.045053	-0.063290319	-0.068922429
Unghiul de rotire al mâinii	67.94713	86.29189301	100.3051834
Clasă gest	4	5	6

Tabelul 5.2 Exemplu valori colectate pentru gestul de rotire al palmei

Un total de 9345 de date au fost colectate pentru fiecare variabilă din cele 29 de categorii ce descriu gesturile. Datele colectate pot fi consultate la următoarea adresă:

https://uptro29158-my.sharepoint.com/:t:/g/personal/stelian_nicola_upt_ro/ERxsvp20MBJIuzIm6cvdAy4BhlYm5Mzrefgv2mmXlW7XLw?e=h5oIyq

Pentru gestul de flexie și extensie al mâinii s-a realizat un al set de date, care dispune de o diversitate mare de valori. Un total de 14 de caracteristici care descriu gestul au fost colectate. La fel ca și la celelalte două gesturi descrise, numărul mare de date ce descriu cele două gesturi au fost selectate pentru a crește acuratețea și precizia de clasificare a gesturilor.

În continuare voi prezenta cel 14 caracteristici colectate:

- O dată ce descrie mâna folosită stânga sau dreapta. Data poate avea valoarea 0 pentru mâna dreaptă sau valoarea 1 pentru mâna stângă.
- 3 date ce descriu rotirea, înclinarea și grația mâinii. Aceste date sunt valori descrise în grade, deoarece reprezintă rotire mâinii față de toate cele 3 axe din spațiul 3D. Toate cele trei date sunt obținute direct prin apelul metodelor specifice: *Roll()*, *Pitch()* și *Yaw()* din biblioteca *Leap*.
- 3 date ce descriu poziția palmei (punctul din mijlocul palmei)
- 3 date ce descriu direcția mâinii (câte o valoare pentru fiecare axă din spațiul 3D)
- 3 date ce descriu direcția degetului mare al mâinii folosite (câte o valoare pentru fiecare axă din spațiul 3D)

Vectorii direcție al mâinii și al degetului mare au fost utilizați pentru a avea o precizie mai mare la detecția mâinii utilizate, stânga sau dreapta.

- O dată ce descrie clasa din care face parte gestul. Aceasta dată este întreagă și poate avea valori cuprinse între 1 și 6. Dacă data are valoare de la 1 la 3 atunci am luat în considerare că mâna dreaptă este folosită, iar dacă are valoare între 4 și 6 mâna stângă este folosită. Fiecare număr reprezintă nivelul în care se găsește gestul.

Datele au fost colectate într-un fișier text a căror valori arată ca și în tabelele 5.3 și 5.4

Mâna stângă sau dreapta	0	0	0
Rotirea mâinii	30.28669803	-120.6368775	-91.97137951
Înclinarea mâinii	-173.6511666	40.50996467	104.514175
Grația mâinii	-11.95881002	166.6060314	95.49417935
Poziție palmă x	-42.16339	-26.41574	-29.69169
Poziție palmă y	204.9274	121.9625	126.32
Poziție palmă z	-44.938	5.21875	-7.040477
Direcția mâinii axa x	-0.197032933	-0.211496107	-0.228479364
Direcția mâinii axa y	0.957642148	0.976485292	0.972040145
Direcția mâinii axa z	-0.209998903	0.041783615	-0.054176901
Direcția degetului mare x	0.147224039	0.354259805	0.319148038
Direcția degetului mare y	0.97478129	0.927926565	0.946510306
Direcția degetului mare z	-0.167709626	0.115983967	-0.04756859
Clasă gest	1	2	3

Tabelul 5.3 Exemplu valori colectate pentru gestul de flexie și extensie pentru mâna dreaptă

Mâna stângă sau dreapta	1	1	1
Rotirea mâinii	96.17132431	28.73097218	87.3663362
Înclinarea mâinii	104.1726382	11.82433811	-101.6018233
Grația mâinii	105.4971212	22.23085531	85.22334386
Poziție palmă x	-25.78506	-31.15907	-37.36585
Poziție palmă y	265.1895	213.369	275.0606
Poziție palmă z	-19.57783	-56.5673	-71.6843
Direcția mâinii axa x	-0.096515991	-0.139771644	-0.130333767
Direcția mâinii axa y	0.992630069	0.957118901	0.959423832
Direcția mâinii axa z	-0.073281714	-0.253746521	-0.250038035
Direcția degetului mare x	0.204648998	0.20281197	0.152802989
Direcția degetului mare y	0.97715559	0.959776327	0.937891233
Direcția degetului mare z	-0.05732138	-0.194156397	-0.311466341
Clasă gest	4	5	6

Tabelul 5.4 Exemplu valori colectate pentru gestul de flexie și extensie pentru mâna stângă

Un total de 2525 de date au fost colectate pentru fiecare variabilă din cele 14 categorii ce descriu gestul de flexie și extensie. Datele colectate pot fi consultate la următoare adresă:
https://uptro29158-my.sharepoint.com/:x:/g/personal/stelian_nicola_upt_ro/EbzoOYlPHN5Hsw_4S6wTlTMBM4JQXcY7mcjN5S10pSMSCQ?e=DDP4Qx

5.2. Filtrarea datelor

Datele colectate au fost filtrate prin două moduri: prin intervale de valori stabilite pentru fiecare clasă de gest și prin număr de valori identice. Aceste două moduri definesc cel mai bine utilitatea gesturilor create. O dată intervalele de valori definesc nivelurile fiecărui gest. Aceste niveluri sunt esențiale pentru viitoare monitorizări ale progresului recuperării mișcărilor/mobilității mâinii. În plus împărțirea pe niveluri ale gesturilor, sunt gândite pentru utilizarea lor de către oamenii ce se află în recuperare.

Pentru cel de-al doilea caz toate datele au fost filtrate astfel încât să nu existe mai mult de 10 valori identice în totalul de tipuri de date. S-a luat această marjă de eroare de maxim 10 valori deoarece pot exista cazuri când două sau mai multe persoane execută gestul definit în același mod și practic este imposibil să se evite valorile duplicate. Întrucât dispozitivul Leap motion face mai mult de 20 de frame-uri pe secundă când este conectat la orice PC există posibilitatea să se înregistreze un

număr ridicat de peste 200 de date identice colectate dacă vorbim despre utilizarea gestului.

Pentru un scenariu în care se colectează date de la minim 10 utilizatori, se pot întâlni minim 200 de date identice ale categoriilor ce descriu gestul de rotire al mâinii sau gestul de închidere sau deschidere al mâinii.

Din punct de vedere tehnic pentru a face aceasta filtrare/restricționare la inserarea datelor în fișierul text s-a folosit proprietatea *Time.DeltaTime* în script-ul C# scris.

Pentru primul mod de filtrare s-au introdus intervale de valori care restricționează colectarea de date ce nu fac parte din acele intervale. Astfel intervalele alese pentru această filtrare se referă la valorile unghiului de rotire al mâinii, dar și al semiperimetrului format de figura geometrică cu coordonate la degetul mare, mijlociu și palma mâinii.

Pentru gestul de rotire al mâinii, mișcarea de pronație și supinație, sunt prezentate următoarele intervale de valori ale unghiului de rotație:

- Unghi = [0; 75)
- Unghi = [75; 100]
- Unghi = (100; 180]

Fiecare interval identifică clasa din care face parte gestul de rotire. Dacă am luat în considerare primul interval atunci gestul este în clasa 4, pentru intervalul al doilea clasa 5, iar pentru ultimul interval clasa 6.

Pentru gestul de strângere și deschidere al mâinii luat în considerare următoarele intervale de valori ale semiperimetrului figurii geometrice cu coordonate la vârfurilor degetului mare, mijlociu și punctul din mijlocul palmei:

- sP = [150; 200]
- sP = (150; 100]
- sP = (100; 0)

Intervalele de mai sus dau clasa în care se găsește gestul de strângere și deschidere al mâinii. Dacă am luat în considerare primul interval atunci gestul este în clasa 1, pentru intervalul al doilea este identificată clasa 2, iar pentru intervalul trei clasa 3 este definită.

5.3. Selectarea modelului rețelei neuronale și clasificarea datelor

Pentru clasificarea datelor colectate și a avea o precizie cât mai bună în detecția celor trei gesturi descrise în subcapitolele precedente, am folosit două rețele neuronală convoluționale pe care am aplicat pe una dintre acestea 11 modele, iar pe alta un singur model. Fiecare rețea este specifică folosirii acestor modele primele 11 sunt aplicate pe prima rețea, iar ultimul model, DNN pe a doua. În continuare vor fi enumerate modelele utilizate împreună cu abrevierile specifice fiecăruia, a căror definire și exemplificare se regăsește în capitolul 3:

- Logistic Regression – LR (regresie logistică)
- Linear Discriminant Analysis - LDA (Analiză discriminantă liniară)

- Kneighbors Classifier – KNN (Clasificator k de vecini)
- Decision Tree Classifier - CART (Clasificator de arbore de decizie)
- Gaussian Naive Bayes – NB (Clasificator Gaussian Naive Bayes)
- SVC1 (Clasificator suport vector 1) - SVC
- SVC2 (Clasificator suport vector 2) - Liniar SVM
- SVM (Mașină vectorială de suport) - RBF SVM
- Random Forest Classifier (Clasificator de arbori aleatoriu) - Random Forest
- Ada Boost Classifier (Clasificator Ada Boost) - AdaBoost
- MLPClassifier (Clasificator de perceptron multistrat) – MLP
- DNNClassifier (Clasificator cu model de rețea neuronală cu învățare profundă)

```
url ='C:/Users/Admin/Desktop/Articol ICIMTH/Date articol/asd/Date gest.txt'

col_name = ['T_x','T_y','T_z','I_x','I_y','I_z','M_x','M_y','M_z','R_x','R_y','R_z','P_x','P_y','P_z','T_I','T_M','T_R','T_p','T_P','I_P','M_P','R_P','P_P','d_x_T','d_y_T','d_z_T','angle','Class']

dataset = pd.read_csv(url, names = col_name)

X = dataset.drop(['Class'], axis=1)
y = dataset['Class']
print(f'X shape: {X.shape} | y shape: {y.shape} ')

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=1)

models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVC', SVC(gamma='auto')))
models.append(('Linear SVM', SVC(kernel="linear", C=0.025)))
models.append(('RBF SVM', SVC(gamma=2, C=1)))
models.append(('Random Forest', RandomForestClassifier(max_depth=5, n_estimators=10, max_features=1)))
models.append(('AdaBoost', AdaBoostClassifier()))
models.append(('MLP',MLPClassifier(alpha=1, max_iter=1000)))
results = []
model_names = []
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
    cv_results = cross_val_score(model, X_train, y_train, cv=kfold, scoring='accuracy')
    results.append(cv_results)
    model_names.append(name)
    print('%s: %f' % (name, cv_results.mean()))
```

Figura 5.5 Script Python de clasificare a gestului de strângere și închidere a mâinii și a gestului de rotire a palmei – CNN 1

În scriptul Python din figura 5.5 este exemplificat pe prima rețea neuronală folosită pentru utilizarea primelor 11 modele. Această rețea este specifică bibliotecii Python, *sklearn.* Pentru început se ia calea spre datele despre gesturi, apoi se identifică fiecare coloană ce descrie o caracteristică a gestului. Pentru a avea o contorizare cât mai bună a numărului de rânduri din fișierul citit am printat pe ecran numărul acestora. Variabilele *X_train* și *y_train, precum și X_test* și *y_test* sunt folosite pentru datele de antrenare și testare a rețelei neuronale. Datele au fost împărțite sub următoarele proporții: 80% sunt date de antrenare și 20% sunt date de testare. Aceste date sunt luate într-un mod aleator din același fișier text de date ce descriu gesturile 1 și 2.

Pentru fiecare dintre cele 11 modele definite din lista de modele se calculează acuratețea rețelei neuronale.

```
def main(argv):
    args = parser.parse_args(argv[1:])
    (train_x, train_y), (test_x, test_y) = niv_gest_rot.load_data()
    print(train_x)
    print(train_y)
    my_feature_columns = []
    for key in train_x.keys():
        my_feature_columns.append(tf.feature_column.numeric_column(key=key))
    classifier = tf.estimator.DNNClassifier(
        feature_columns=my_feature_columns,
        hidden_units=[16, 16],
        n_classes=7)
    classifier.train(
        input_fn=lambda:niv_gest_rot.train_input_fn(train_x, train_y,
                                                    args.batch_size),
        steps=args.train_steps)
    eval_result = classifier.evaluate(
        input_fn=lambda:niv_gest_rot.eval_input_fn(test_x, test_y,
                                                   args.batch_size))
    predictionss = classifier.predict(
        input_fn=lambda:niv_gest_rot.eval_input_fn(test_x, test_y,
                                                   args.batch_size))
    pred=[]
    for pred_dicts, expecs in zip(predictionss, test_y):
        class_ids = pred_dicts['class_ids'][0]
        pred.append(class_ids)
    print(metrics.classification_report(test_y, pred))
    print('\nTest acuratete: {acuratetea:0.3f}\n'.format(**eval_result))
```

Figura 5.6 Script Python de clasificare a gestului de strângere și închidere a mâinii și a gestului de rotire a palmei – CNN 2

Script-ul Python din figura 5.6 exemplifică cea dea doua rețea folosită. Aceasta este definită în biblioteca *tensorflow.* Modelul aplicat pe această rețea neuronală convoluțională este DNN (model de rețea neuronală profundă). Pentru început datele au fost împărțite în date de antrenare și date de test prin folosirea variabilelor *train_x, train_y, test_x și test_y.* Mai departe se calculează numărul de straturi ascunse din rețea și se aplică modelul DNN. Numărul de straturi ascunse s-a calculat pe baza formulelor exemplificate în capitolul 3. Pentru acest model s-a utilizat 32 de straturi ascunse. După aplicarea clasificatorului DNN, se indexează fiecare clasă a gesturilor de la 0 la 6, apoi se printează acuratețea modelului în clasificarea gesturilor 1 și 2. Dacă la prima rețea datele au fost împărțite în 80% date de antrenare și 20% date de test, pentru această rețea datele de antrenare au fost luate dintr-un fișier text separat față de datele de antrenare.
În tabelul 5.7 sunt prezentate rezultatele obținute pentru valorile acurateței modelelor folosite:

	Acuratețe
LR	0.94
LDA	0.91
KNN	0.98
CART	0.99
NB	0.84
SVC	0.88

Liniar SVM	0.99
RBF SVM	0.69
Random Forest	0.91
AdaBoost	0.60
MLP	0.96
DNN	0.47

Tabel 5.7 Acuratețea modelelor la clasificarea gesturilor 1 și 2

Din tabelul 5.7 pe primele locuri în clasificarea gesturilor din punct de vedere al acurateței se regăsesc modelele: Liniar SVM, CART și KNN, iar pe ultimele locuri se regăsesc modelele: DNN, AdaBoost și RBF SVM.

Model	**Acuratețe**
LR	0.94
LDA	0.91
KNN	0.96
CART	0.99
NB	0.96
SVC	0.51
Liniar SVM	0.97
RBF SVM	0.45
Random Forest	0.89
AdaBoost	0.99
MLP	0.96
DNN	0.95

Tabel 5.8 Acuratețea modelelor la clasificarea gestului 3

Din tabelul 5.8 pe primele locuri în clasificarea gesturilor din punct de vedere al acurateței se regăsesc modelele: CART, AdaBoost și Liniar SVM, iar pe ultimele locuri se regăsesc modelele: RBF SVM, SVC și Random Forest.
Tabelele 5.7 și 5.8 exprimă rezultatul acuratețelor de clasificare a gesturilor 1, 2 și 3 pentru ambele rețele neuronale folosite împreună cu modelele/clasificatoarele specifice acestora. Script-urile Python exemplificate pentru fiecare rețea au fost aplicate pe rând pe fiecare set de date care descriu o dată primele două gesturi, apoi ultimul gest.
Cele mai importante biblioteci python folosite sunt: tensorflow, *sklearn, numpy, pandas, matplotlib și seaborn*.

5.4. Metrici – precizie, recall, f1 scor

În tabelul 5.9 sunt prezentate principalele valori ale metricilor, obținute pe fiecare model pentru clasificarea gesturilor de recuperare (primele două gesturi descrise în acest capitol), gestul de închidere și deschidere al mâinii, precum și gestul de rotire al mâinii. Modul de calcul al metricilor: precizie, recall și f1 scor, este exemplificat în capitolul 3.

Model	**Clasă**	**Precizie**	**Recall**	**F1-scor**	**Număr date**
DNN	1	0.20	0.96	0.33	51

	2	0.96	0.89	0.92	61
	3	0.89	1.00	0.94	64
	4	0.00	0.00	0.00	56
	5	1.00	0.23	0.37	75
	6	0.00	0.00	0.00	83
LR	1	0.95	0.94	0.95	788
	2	0.97	0.96	0.97	288
	3	0.99	1.00	1.00	529
	4	0.82	0.91	0.86	175
	5	0.86	0.76	0.81	33
	6	0.88	0.82	0.85	56
LDA	1	0.97	0.86	0.91	788
	2	0.99	0.91	0.95	288
	3	0.97	1.00	0.99	529
	4	0.73	0.97	0.83	175
	5	0.45	0.85	0.59	33
	6	0.74	0.86	0.79	56
KNN	1	0.99	0.99	0.99	788
	2	0.99	1.00	0.99	288
	3	1.00	1.00	1.00	529
	4	0.96	0.98	0.97	175
	5	1.00	0.88	0.94	33
	6	0.98	0.95	0.96	56
CART	1	1.00	0.99	1.00	788
	2	0.99	0.99	0.99	288
	3	0.99	1.00	1.00	529
	4	0.97	1.00	0.98	175
	5	1.00	0.94	0.97	33
	6	0.98	0.98	0.98	56
NB	1	0.98	0.73	0.84	788
	2	0.73	0.93	0.82	288
	3	0.97	0.92	0.95	529
	4	0.70	0.95	0.81	175
	5	0.51	0.91	0.65	33
	6	0.48	0.98	0.65	56
SVC	1	0.81	1.00	0.89	788
	2	1.00	0.71	0.83	288
	3	1.00	0.91	0.95	529
	4	0.99	0.90	0.94	175
	5	1.00	0.36	0.53	33
	6	1.00	0.75	0.86	56
Liniar SVM	1	0.99	0.99	0.99	788
	2	0.99	1.00	1.00	288
	3	1.00	1.00	1.00	529
	4	0.98	0.99	0.99	175
	5	1.00	0.91	0.95	33
	6	0.98	0.98	0.98	56
RBF SVM	1	0.58	1.00	0.73	788
	2	1.00	0.44	0.61	288
	3	1.00	0.52	0.69	529
	4	1.00	0.38	0.55	175
	5	1.00	0.30	0.47	33
	6	1.00	0.38	0.55	56
Random Forest	1	0.84	1.00	0.92	788
	2	0.98	0.93	0.95	288

	3	0.98	0.99	0.99	529
	4	1.00	0.56	0.72	175
	5	1.00	0.85	0.92	33
	6	1.00	0.07	0.13	56
AdaBoost	1	0.80	0.91	0.85	788
	2	0.20	0.47	0.28	288
	3	0.00	0.00	0.00	529
	4	0.61	0.96	0.75	175
	5	0.86	0.55	0.67	33
	6	1.00	0.32	0.49	56
MLP	1	0.96	0.98	0.97	788
	2	0.99	0.98	0.99	288
	3	1.00	1.00	1.00	529
	4	0.94	0.94	0.94	175
	5	0.97	0.85	0.90	33
	6	1.00	0.77	0.87	56

Tabelul 5.9 Metrici modele de clasificare a gesturilor strângere și deschidere al palmei, rotire al palmei

În tabelul 5.10 sunt prezentate principalele valori ale metricilor, obținute pentru clasificarea gesturilor de recuperare (ultimul gest descris în acest capitol), gestul de flexie și extensie al mâinii pentru fiecare nivel al acestuia.

Model	Clasă	Precizie	Recall	F1-scor	Număr date
DNN	1	0.94	0.99	0.96	152
	2	0.99	0.92	0.95	146
	3	0.41	1.00	0.59	12
	4	0.88	0.73	0.80	52
	5	1.00	0.96	0.98	368
	6	0.86	0.98	0.92	56
LR	1	0.95	0.95	0.95	62
	2	0.98	0.92	0.95	113
	3	0.90	0.83	0.86	46
	4	0.77	0.89	0.82	37
	5	0.97	0.99	0.98	218
	6	0.93	0.97	0.95	29
LDA	1	1.00	0.82	0.90	62
	2	0.89	0.96	0.92	113
	3	0.90	0.76	0.82	46
	4	0.76	0.84	0.79	37
	5	0.97	0.98	0.97	218

	6	0.90	0.97	0.93	29
	1	0.98	0.97	0.98	62
	2	0.97	0.98	0.97	113
	3	0.90	0.93	0.91	46
KNN	4	0.97	0.76	0.85	37
	5	0.97	1.00	0.99	218
	6	1.00	0.97	0.98	29
	1	1.00	1.00	1.00	62
	2	1.00	1.00	1.00	113
	3	1.00	1.00	1.00	46
CART	4	1.00	1.00	1.00	37
	5	1.00	1.00	1.00	218
	6	1.00	1.00	1.00	29
	1	0.98	0.97	0.98	62
	2	0.98	0.97	0.98	113
	3	0.96	1.00	0.98	46
NB	4	0.73	1.00	0.84	37
	5	1.00	0.91	0.95	218
	6	0.85	1.00	0.92	29
	1	1.00	0.29	0.45	62
	2	1.00	0.18	0.30	113
	3	1.00	0.33	0.49	46
SVC	4	1.00	0.11	0.20	37
	5	0.49	1.00	0.66	218
	6	1.00	0.17	0.29	29
	1	1.00	0.98	0.99	62
	2	0.98	0.96	0.97	113
	3	1.00	0.96	0.98	46
Liniar SVM	4	0.95	1.00	0.97	37
	5	0.98	1.00	0.99	218
	6	1.00	1.00	1.00	29
	1	1.00	0.05	0.09	62
	2	1.00	0.07	0.13	113
RBF SVM	3	1.00	0.04	0.08	46
	4	1.00	0.03	0.05	37

Random Forest	5	0.44	1.00	0.62	218
	6	1.00	0.03	0.07	29
	1	0.98	0.81	0.88	62
	2	0.83	0.94	0.88	113
	3	0.96	0.52	0.68	46
	4	1.00	0.51	0.68	37
	5	0.85	1.00	0.92	218
	6	1.00	0.93	0.96	29
AdaBoost	1	1.00	1.00	1.00	62
	2	1.00	1.00	1.00	113
	3	1.00	1.00	1.00	46
	4	1.00	1.00	1.00	37
	5	1.00	1.00	1.00	218
	6	1.00	1.00	1.00	29
MLP	1	0.98	0.97	0.98	62
	2	0.99	0.95	0.97	113
	3	0.94	1.00	0.97	46
	4	0.95	0.97	0.96	37
	5	1.00	1.00	1.00	218
	6	0.97	1.00	0.98	29

Tabelul 5.10 Metrici model de clasificare a gestului de flexie și extensie

Pentru a vizualiza mai bine rezultatele obținute în urma clasificărilor pentru gestul 1 și 2 prin cele 12 modele s-a creat pentru fiecare model o matrice de confuzie.

În continuare va fi prezentat scriptul aferent matricelor de confuzie împreună cu rezultatul acestuia.

```
model = LogisticRegression()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - LR')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

Figura 5.11 Script matrice confuzie pentru modelul LR

În figura 5.11 este exemplificat script-ul Python aferent matricei de confuzie pentru modelul LR. Variabila *model* este inițializată cu funcția *LogisticRegression(),* apoi sunt inițializate *prediction,* proprietatea *model.fit* și variabila *cm* cu datele aferente fiecăreia pentru construirea matricei de confuzie. Sunt marcate liniile și coloanele cu valorile actuale și respectiv cele prezise, iar apoi se asociază la intersecția fiecărui element din matrice cu numărul de valori prezise.

Script-urile python aferente celorlalte modele ale matricelor de confuzie sunt asemănătoare cu acesta, modificarea care se face fiind funcția ce reprezintă modelul aplicat pe rețeaua neuronală și numele modelului aplicat (LR, LD, SMV, AdaBoost) în matricea de confuzie. Variabila *model* este inițializată pe rând cu fiecare funcție ce reprezintă modelul aplicat. Astfel sunt prezente următoarele funcții:

- *LogisticRegression()*
- *LinearDiscriminantAnalysus()*
- *KNeighborsClassifier()*
- *DecisionTreeClassifier()*
- *GausianNB()*
- *SVC(gamma='auto')*
- *SVC(kernel='linear',C=0.025)*
- *SVC(gamma=2,C=1)*
- *RandomForestClassifier(max_depth=5,n_estimators=10,max_features=1)*
- *AdaBoostClassifier()*
- *MLPClassifier(alpha=1,max_iter=1000)*
- *DNNClassifier(feature_columns=my_feature_columns, hidden_units=[16, 16], n_classes=7)*

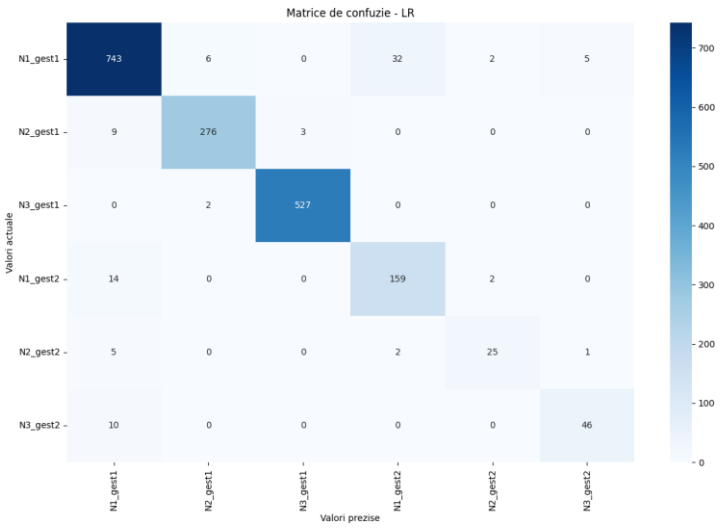

Figura 5.12. Matricea de confuzie pentru modelul LR ale gesturilor 1 și 2

Figura 5.12 prezintă matricea de confuzie pentru modelul LR. Pentru gestul 1, nivel 1 45 de cazuri au fost prezise greșit, dintr-un total de 788 de teste. Nivelul 2

al gestului 1 a fost prezis greșit de 12 ori dintr-un total de 288 de cazuri, iar nivelul 3 al gestului 1 a fost prezis de 2 ori greșit dintr-un total de 529 de teste. Pentru gestul 2, nivelul 1 a fost detectat corect de 159 de ori dintr-un total de 175 de cazuri. Se observă că acesta a fost confundat de 14 de ori cu nivelul 1 al gestului 1 și de 2 ori cu nivelul 2 al gestului 2. Nivelul 2 al gestului 2 a fost prezis de 25 de ori corect dintr-un total de 33 de cazuri, iar nivelul 3 al gestului 2 a fost prezis corect de 46 de ori dintr-un total de 56 de cazuri.

```
model = LinearDiscriminantAnalysis()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - LDA')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

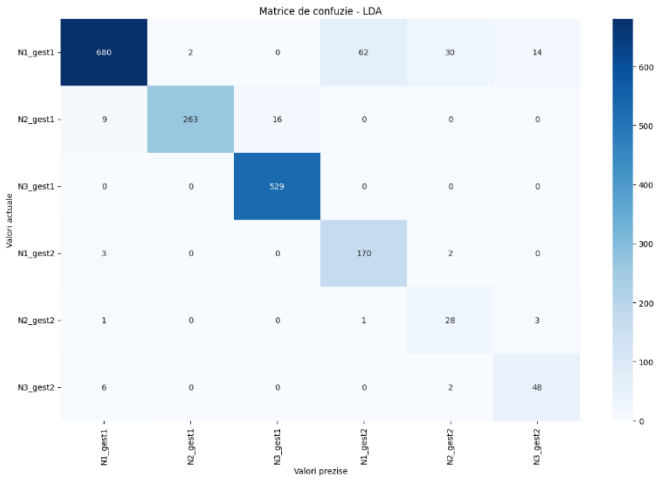

Figura 5.13. Matricea de confuzie pentru modelul LDA ale gesturilor 1 și 2

Figura 5.13 prezintă matricea de confuzie pentru modelul LDA. Gestul 1, nivelul 1 a fost prezis greșit de 108 ori dintr-un total de 788 de teste pentru acesta. Nivelul 2 al gestului 1 a fost prezis greșit doar de 25 ori dintr-un total de 288 de teste. Nivelul 3 al gestului 1 a fost prezis perfect în toate cele 529 de cazuri. Pentru gestul al doilea nivelul1 a fost prezis greșit de 5 ori dintr-un total de 175 de cazuri de testare. Nivelul 2 a fost prezis corect în 28 de cazuri dintr-un total de 33 de cazuri de testare. Nivelul 3 al gestului 2 a fost prezis greșit în 8 cazuri dintr-un total de 56 de teste. Acesta a fost confundat cu nivelul 1 al gestului 1 și cu nivelul 2 al gestului 2.

```
model = KNeighborsClassifier()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - KNN')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

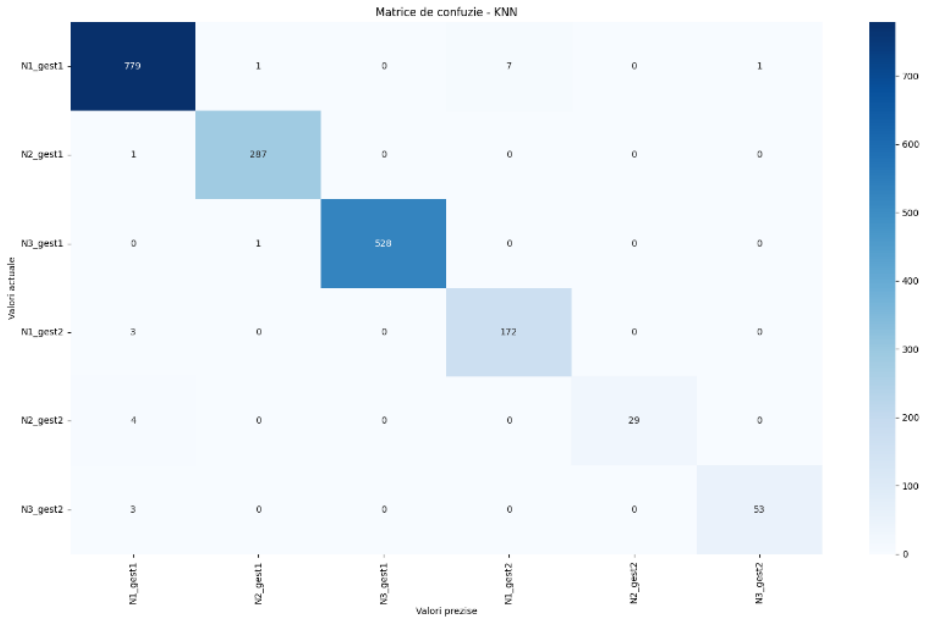

Figura 5.14. Matricea de confuzie pentru modelul KNN ale gesturilor 1 și 2

Din matricea de confuzie din figura 5.14 se observă o detecție corectă în proporții de aproape 100% pentru nivelurile 3 și 2 ale gestului 1. O singură dată pentru fiecare test pe nivel s-a greșit. Nivelul 1 al gestului 1 a fost detectat corect în 779 din cazuri, dintr-un total de 788 de teste. Nivelul 2 al gestului 1 a fost detectat greșit de o dată dintr-un total de 288 de teste. Nivelul 1 al gestului 2 a fost detectat corect în 172 de cazuri, dintr-un total 175 de teste, iar nivelul 3 gestului 2 a fost detectat corect în 53 de cazuri dintr-un total de 56 de teste. Se poate spune că nu există o legătură dintre numărul de teste și detecția corectă a nivelului gestului deoarece chiar dacă sunt puține cazuri la nivelul 2 al gestului 2, un total de 29.

Totodată pentru nivelul 3 al gestului 1 sunt 528 de teste care au avut o detecție corectă aproape de 100%.

```
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - CART')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

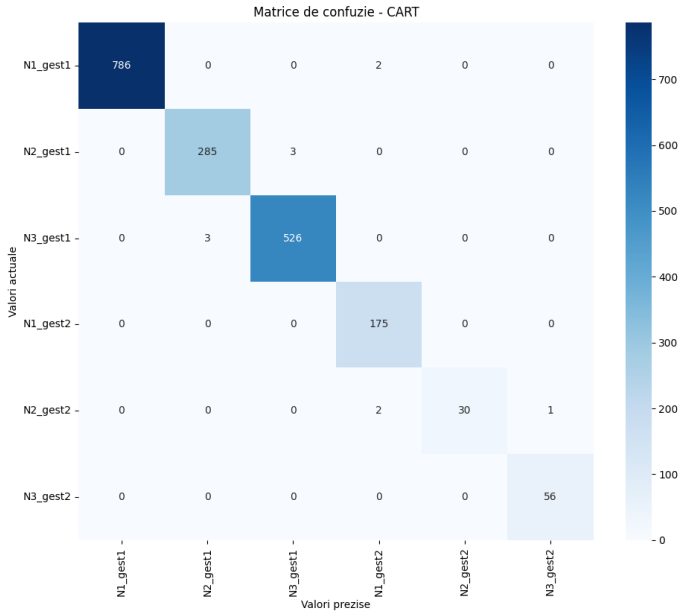

Figura 5.15. Matricea de confuzie pentru modelul CART ale gesturilor 1 și 2

Matricea de confuzie al modelului CART este prezentată în figura 5.15. Nivelurile 2 și 3 ale gestului 1 au fost detectate greșit fiecare în 3 cazuri dintr-un total de 288, respectiv 529 de teste. Nivelul 1 al gestului 2 a fost detectat corect în 175 de cazuri dintr-un total de 175, iar nivelul 3 al gestului 2 a fost detectat corect în 56 de cazuri dintr-un total de 56 de teste. Din nou se observă lipsa de legătură din numărul de teste și precizia de detecție, deoarece nivelul 1 al gestului 2 a fost detectat corect în toate 175 de cazuri, iar nivelul 3 al gestului 2 a fost detectat corect în toate cele 56 de cazuri.

```
model = GaussianNB()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - NB')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

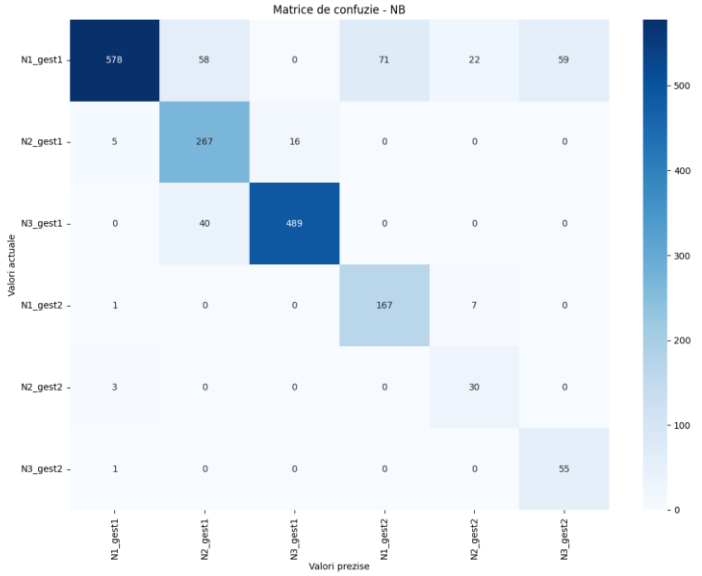

Figura 5.16. Matricea de confuzie pentru modelul NB ale gesturilor 1 și 2

Figura 5.16 prezintă matricea de confuzie pentru modelul NB. Nivelul 1 al gestului 1 a fost detectat corect în 578 de cazuri dintr-un total de 788 de teste. Nivelul 2 și 3 al înregistrat o performanță mai ridicată, fiind detectate greșit în 61 de cazuri dintr-un total de 817 cazuri de testare. Pentru gestul 2 nivelul 1 a fost detectat corect în 167 de cazuri dintr-un total de 175 de teste. Nivelul 2 a fost detectat corect în 30 de cazuri dintr-un total de 33 de teste. Nivelul 3 al gestului 2 a fost confundat cu nivelul 1 al gestului 1 într-un singur caz dintr-un total de 56 de teste.

```
model = SVC(gamma='auto')
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - SVC')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

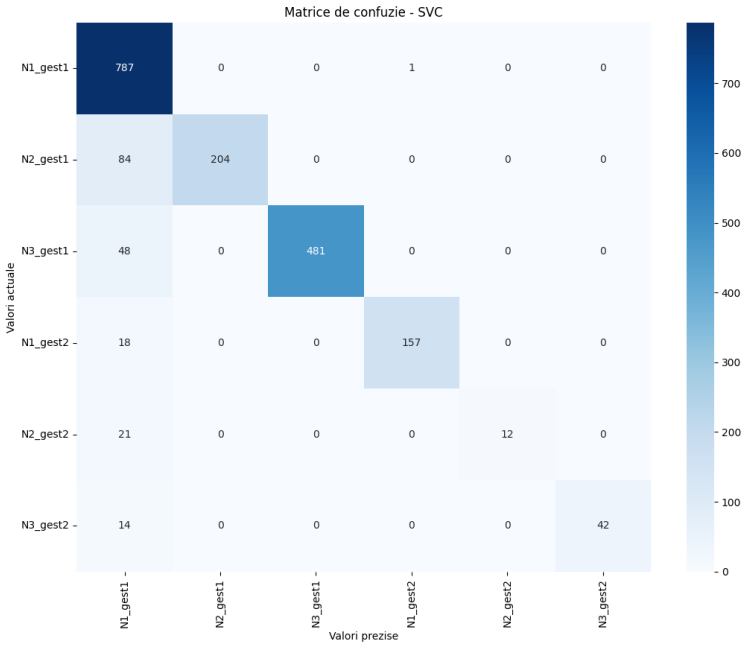

Figura 5.17. Matricea de confuzie pentru modelul SVC ale gesturilor 1 și 2

Matricea de confuzie al modelului SVC pentru gestul1 și 2 din figura 5.17. Nivelul 2 și 3 al gestului 1 a fost detectat corect în 685 de cazuri dintr-un total de 817 de cazuri de testare. Nivelul 2 și 3 al gestului 2 a fost detectat corect în 54 de cazuri de testare dintr-un total de 89 de teste. Nivelul 1 al gestului 2 a fost detectat corect în aproape toate cazurile. Doar în18 cazuri a fost detectat greșit dintr-un total de 175 de cazuri de testare. Nivelul 1 al gestului 1 a fost detectat corect în aproape toate cazurile, doar o singură dată a fost confundat cu nivelul 1 al gestului 2 dintr-un total de 788 de cazuri de testare.

```
model = SVC(gamma=2, C=1)
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - Liniar SVM')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

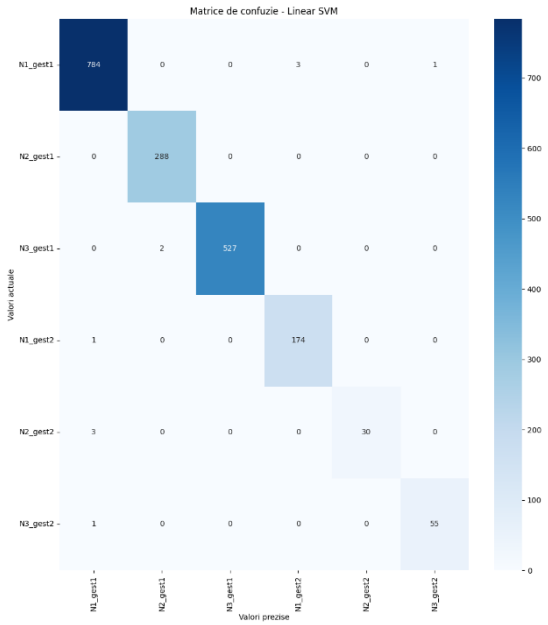

Figura 5.18. Matricea de confuzie pentru modelul Liniar SVM ale gesturilor 1 și 2

Figura 5.18 prezintă matricea de confuzie pentru modelul Liniar SVM. Nivelul 1 al gestului 1 a fost detectat corect în 784 de cazuri dintr-un total de 788 de teste. Nivelul 2 a fost detectat corect în 288 de cazuri dintr-un total de 288 de teste. Nivelul 3 al gestului 1 a fost detectat corect în 527 de cazuri dintr-un total de 529 de teste. Nivelul 2 al gestului 2 a fost detectat corect în 30 de cazuri din 33. Pentru nivelul 1 și 3 al gestului 2 au fost detectate corect 229 de cazuri dintr-un total de 231 de cazuri de testare.

```
model = SVC(gamma=2, C=1)
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - RBF SVM')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

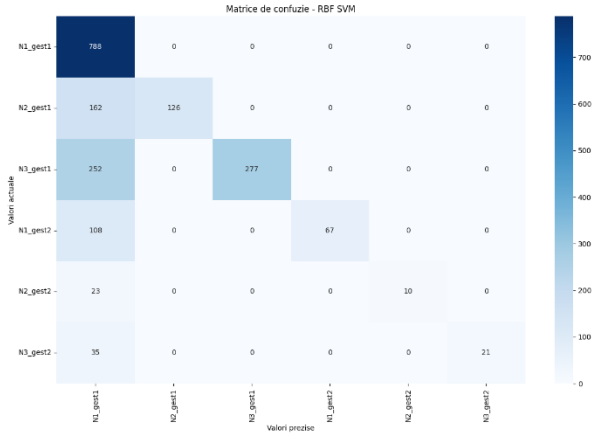

Figura 5.19. Matricea de confuzie pentru modelul RBF SVM ale gesturilor 1 și 2

Figura 5.19 prezintă matricea de confuzie pentru modelul RBF SVM. Gestul 1, nivel 1 a fost detectat corect în toate cele 788 de cazuri de testare. Nivelul 2 al gestului 1 a fost prezis greșit de 162 de ori dintr-un total de 288 de cazuri, iar nivelul 3 al gestului 1 a fost prezis de 252 ori greșit dintr-un total de 529 de teste. Pentru gestul 2, nivelul 1 a fost detectat corect de 67 de ori dintr-un total de 175 de cazuri. Se observă că acesta a fost confundat de 108 ori cu nivelul 1 al gestului 1. Nivelul 2 al gestului 2 a fost prezis de 10 ori corect dintr-un total de 33 de cazuri, iar nivelul 3 al gestului 2 a fost prezis corect de 21 de ori dintr-un total de 56 de cazuri.

```
model = RandomForestClassifier(max_depth=5, n_estimators=10, max_features=1)
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - Random Forest')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

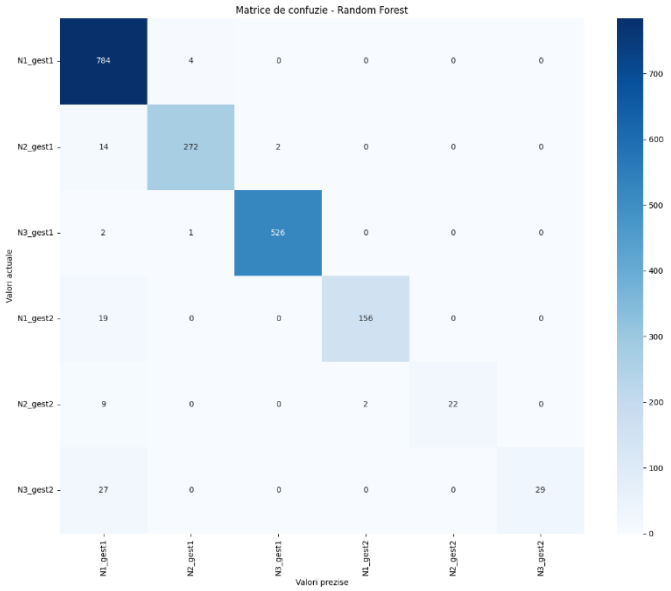

Figura 5.20. Matricea de confuzie pentru modelul Random Forest ale gesturilor 1 și 2

Figura 5.20 prezintă matricea de confuzie pentru modelul Random Forest. Pentru gestul 1, nivel 1, 4 cazuri au fost prezise greșit, dintr-un total de 788 de teste. Nivelul 2 al gestului 1 a fost detectat greșit de 16 ori dintr-un total de 288 de cazuri, iar nivelul 3 al gestului 1 a fost prezis de 3 ori greșit dintr-un total de 529 de teste. Pentru gestul 2, nivelul 1 a fost detectat corect de 156 de ori dintr-un total de 175 de cazuri. Nivelul 2 și 3 al gestului 2 a fost detectat corect în 51 de cazuri dintr-un total de 89 cazuri de testare. Se observă că numărul de teste nu este legat de precizia de detecție al modelului Random Forest.

```
model = AdaBoostClassifier()
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - AdaBoost')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

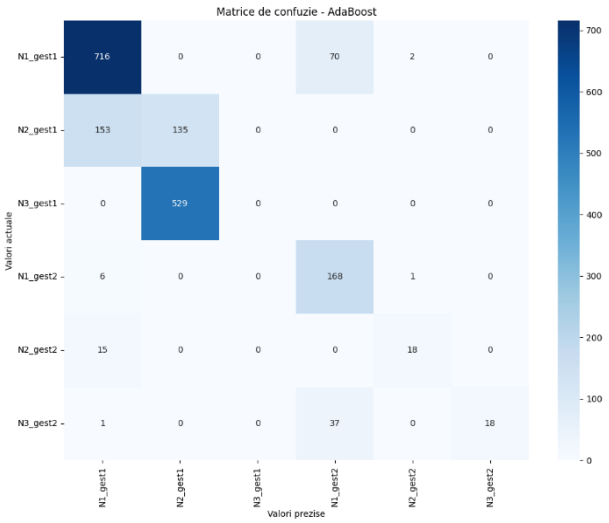

Figura 5.21. Matricea de confuzie pentru modelul AdaBoost ale gesturilor 1 și 2

Matricea de confuzie al modelului AdaBoost ale gesturilor 1 și 2 este prezentată în figura 5.21. Se observă un hazard legat de numărul de cazuri testare pentru fiecare nivel al gesturilor 1 și 2. Astfel pentru nivelul 3 al gestului 1 nu a fost detectat corect niciodată, iar pentru nivelurile 2 și 3 ale gestului 2 s-au realizat 89 de teste, fiind detectate corect în 36 de cazuri. Nivelul 1 al gestului 2 a fost confundat de 70 de ori cu nivelul 1 al gestului 1 și de 2 ori cu nivelul 2 al gestului 2 dintr-un total de 175 de teste. Nivelul 1 al gestului 1 a fost detectat corect în 716 cazuri dintr-un total de 788 de cazuri de testare. De cele mai multe ori acesta a fost confundat cu nivelul 1 al gestului 2. Pentru nivelul 2 al gestului 1 este evidențiată o confuzie mare între acesta și nivelul 1 al aceluiași gest. Astfel în 153 de cazuri acesta a fost confundat cu nivelul 1 dintr-un total de 288 de cazuri de testare.

```
model = MLPClassifier(alpha=1, max_iter=1000)
model.fit(X_train, y_train)
prediction = model.predict(X_test)
cm = confusion_matrix(y_test, prediction)
cm_df = pd.DataFrame(cm,
                     index = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'],
                     columns = ['N1_gest1','N2_gest1','N3_gest1','N1_gest2','N2_gest2','N3_gest2'])

plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,fmt='d',cmap='Blues')
plt.title('Matrice de confuzie - MLP')
plt.ylabel('Valori actuale')
plt.xlabel('Valori prezise')
plt.show()
```

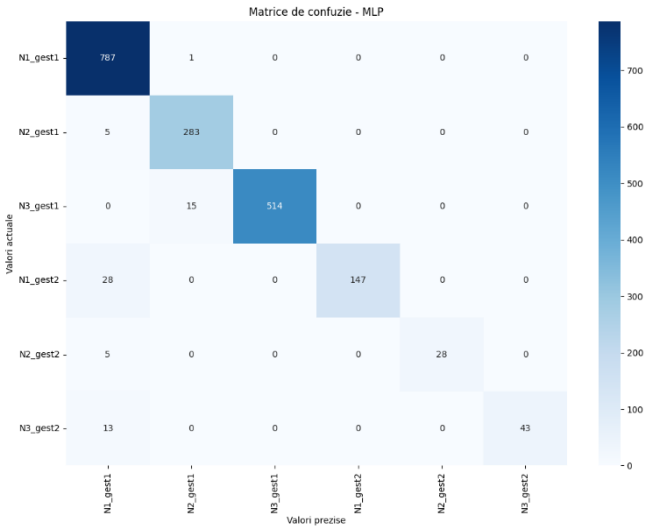

Figura 5.22. Matricea de confuzie pentru modelul MLP ale gesturilor 1 și 2

Figura 5.22 prezintă matricea de confuzie pentru modelul MLP. Pentru gestul 1, nivel 1, într-un singur caz fost prezis greșit, dintr-un total de 788 de teste. Nivelul 2 al gestului 1 a fost prezis greșit de 5 ori dintr-un total de 288 de cazuri, iar nivelul 3 al gestului 1 a fost prezis de 15 ori greșit dintr-un total de 529 de teste. Pentru gestul 2, nivelul 1 a fost detectat corect de 147 de ori dintr-un total de 175 de cazuri. Se observă că acesta a fost confundat de 28 de ori cu nivelul 1 al gestului 1. Nivelul 2 al gestului 2 a fost prezis de 28 de ori corect dintr-un total de 33 de cazuri, iar nivelul 3 al gestului 2 a fost prezis corect de 43 de ori dintr-un total de 56 de cazuri.

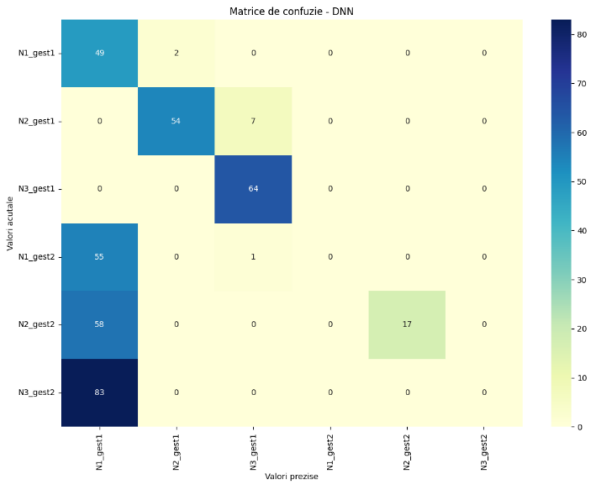

Figura 5.23. Matricea de confuzie pentru modelul DNN ale gesturilor 1 și 2

Figura 5.23 prezintă matricea de confuzie pentru modelul DNN ale gesturilor 1 și 2. Se observă că cele mai slabe rezultate le prezintă gestul al doilea pentru nivelul 1 și nivelul 3, chiar dacă pentru aceste nivele s-a testat cu un număr de 139 de date de testare. Aceste date nu au fost categorisite în nici un nivel din celelalte gesturi. Nivelul 2 al gestului 2 a avut o precizie de 100%, în toate cel 17 cazuri acesta a fost detectat corect. Nivelul 2 și 3 al primului gest prezintă rezultate bune, acestea fiind detectate corect în 118 cazuri dintr-un total de 128 de cazuri de testare. Pentru nivelul 1 al gestului 1, din matricea de confuzie se poate observa un hazard în detectarea corectă a acestuia. Astfel dintr-un total de 245 de cazuri de testare, doar 49 dintre acestea au fost detectate corect. În celelalte 196 de cazuri s-a făcut o confuzie cu unul dintre celelalte niveluri ale gestului 2.
În continuare vor fi prezentate matricele de confuzie pentru cele 12 modele utilizate în clasificarea gestului 3, gestul de flexie și extensie al mâinii.

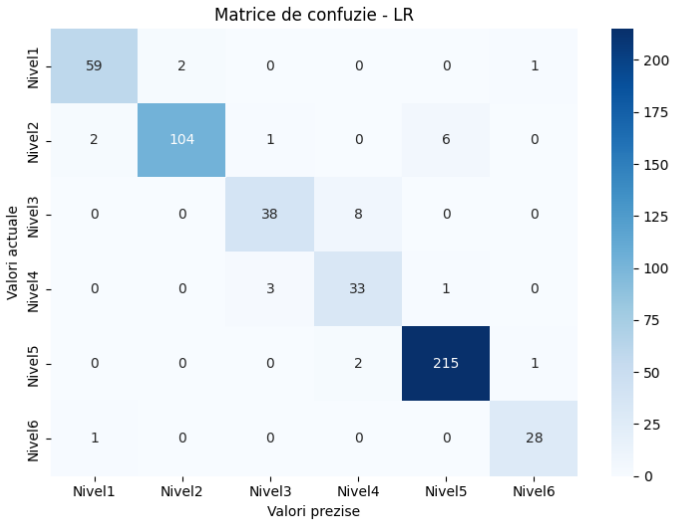

Figura 5.24. Matricea de confuzie pentru modelul LR al gestului 3

Matricea de confuzie al modelului LR pentru gestul 3 din figura 5.24 prezintă rezultatele pentru prezicerea nivelurilor gestului de flexie și extensie. Se observă că pentru nivelul 1, 3 cazuri au fost detectate greșit dintr-un total de 62 de cazuri de testare. Nivelul 1 și 2 a fost detectat greșit în 6 cazuri dintr-un total de 148 de cazuri de testare. Nivelul 4 și 5 al gestului 3 a fost detectat corect în 248 de cazuri dintr-un total de 265 de cazuri de testare. Deși pentru nivelul 6 sunt doar 30 de cazuri de testare acesta a fost detectat corect în 28 de cazuri.

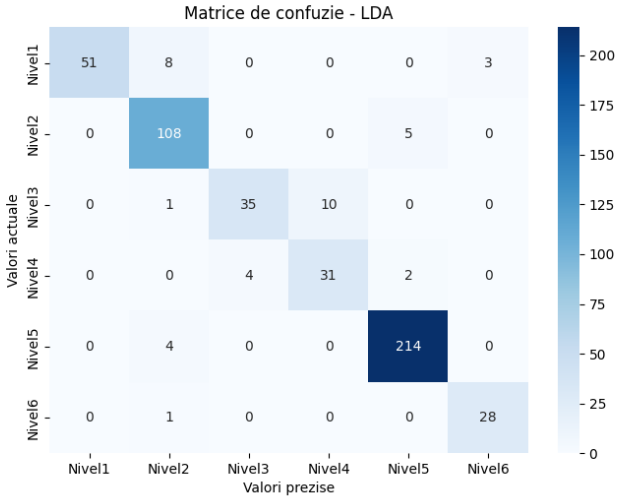

Figura 5.25. Matricea de confuzie pentru modelul LDA al gestului 3

În figura 5.25 este prezentată matricea de confuzie pentru modelul LDA al gestului 3. Se observă că nivelul 1 al gestului a fost detectat corect în toate cele 51 de cazuri de testare. Nivelul 2 și 3 al gestului a fost prezis corect în 143 de cazuri dintr-un total de 161 de teste. Nivelul 5 și 6 al gestului a fost detectat corect în 242 de cazuri, fiind confundate cu celelalte niveluri în 10 cazuri de testare. Pentru nivelul 4 al gestului este prezentată o performanță mai scăzută, 10 cazuri au fost detectate greșit, fiind confundate cu nivelul 3, dintr-un total de 41 de teste.

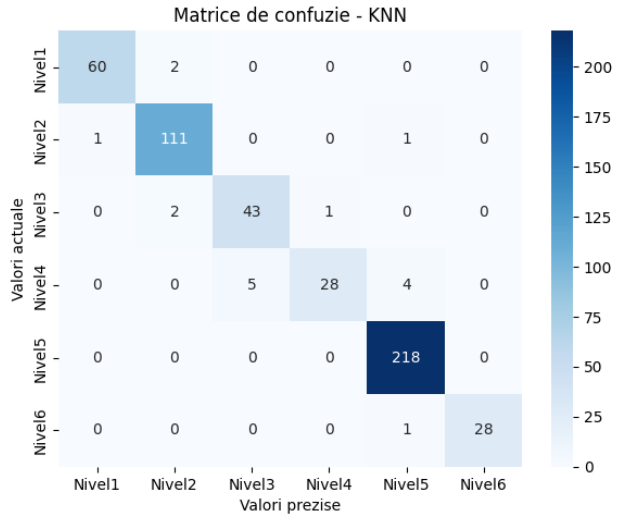

Figura 5.26. Matricea de confuzie pentru modelul KNN al gestului 3

Matricea de confuzie pentru modelul KNN al gestului 3 este prezentată în figura 5.26. Se observă că nivelul 6 al gestului a fost detectate corect în proporții de 100%. Nivelul 1 și 4 are fiecare câte un caz detectat greșit dintr-un total de 90 de cazuri de testare. Nivelul 2 și 3 al gestului au câte 4, respectiv 5 cazuri detectate greșit dintr-un total de 163 de cazuri de testare. În nivelul 5 sunt 218 cazuri detectate corect dintr-un total de 224 de cazuri.

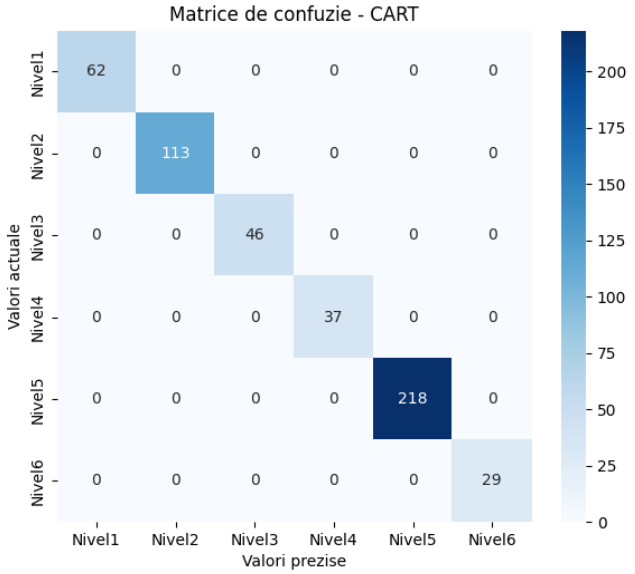

Figura 5.27. Matricea de confuzie pentru modelul CART al gestului 3

Matricea de confuzie pentru modelul CART al gestului 3 prezintă cele mai bune rezultate obținute. Astfel pentru toate cele 505 cazuri detecția nivelurilor este de 100%. Se poate observa că nu există nici o legătură între numărul de teste și performanța de detecție a nivelurilor gestului de flexie și extensie.

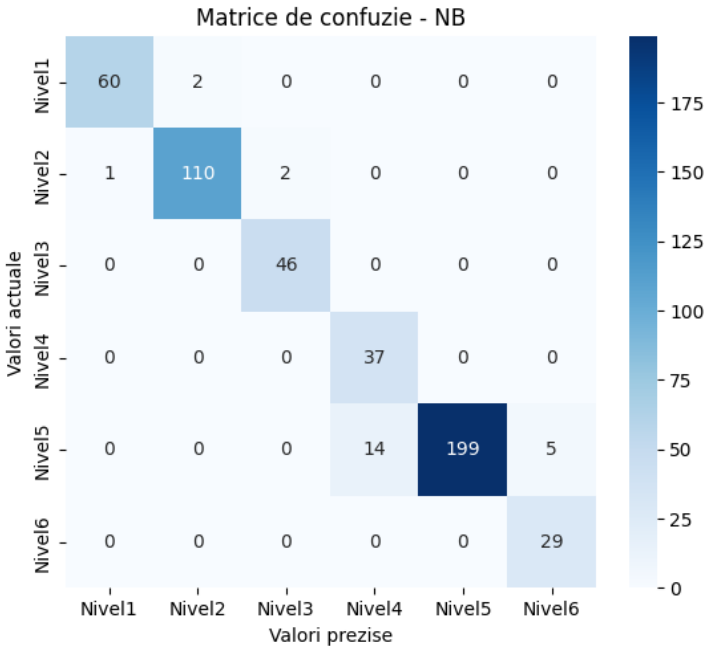

Figura 5.28. Matricea de confuzie pentru modelul NB al gestului 3

Figura 5.28 prezintă matricea de confuzie pentru modelul NB al gestului 3. Nivelul 5 al acestui gest prezintă cele mai bune rezultate. Se observă că acesta a fost prezis corect în toate cele 199 de cazuri de testare. Tot o performanță bună o are și nivelul 1, 2 și 3 al acestui gest. Acestea fiind prezise incorect doar într-un caz pentru nivelul 1 și în câte două cazuri pentru nivelurile 2 și 3. Nivelul 4 a fost prezis corect în 37 de cazuri dintr-un total de 51 de cazuri de testare, iar nivelul 6 a fost prezis corect în 29 de cazuri dintr-un total de 34. Din nou se observă lipsa de legătură între numărul ridicat de cazuri de testare și precizie. Chiar dacă pentru nivelul 5 sunt cele mai multe cazuri de testare, 199, acesta a fost detectat corect în toate din acestea.

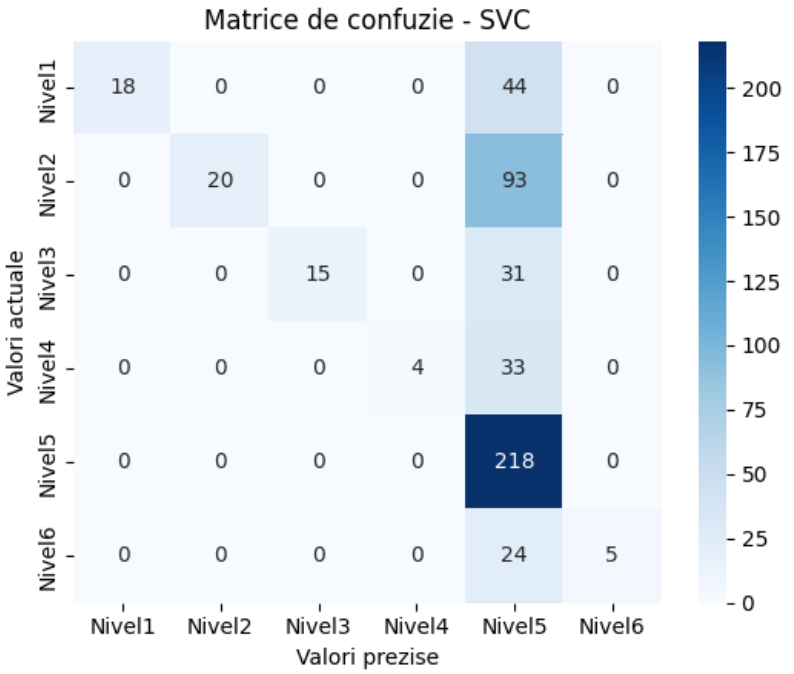

Figura 5.29. Matricea de confuzie pentru modelul SVC al gestului 3

Matricea de confuzie pentru modelul SVC al gestului 3 este prezentată în figura 5.29. Se observă că 5 din cele 6 niveluri ale gestului au o precizie de detectare de 100%. Toate cele 62 de cazuri de testare au fost prezise corect pentru nivelurile 1, 2, 3, 4, 6. Nivelul 5 prezintă cele mai slabe rezultate. Acesta a fost confundat cu celelalte cazuri de 225 de ori dintr-un total de 443 de cazuri de testare. Practic pentru nivelul 5 s-a înregistrat o precizie de detecție a acestuia de aproximativ 50%.

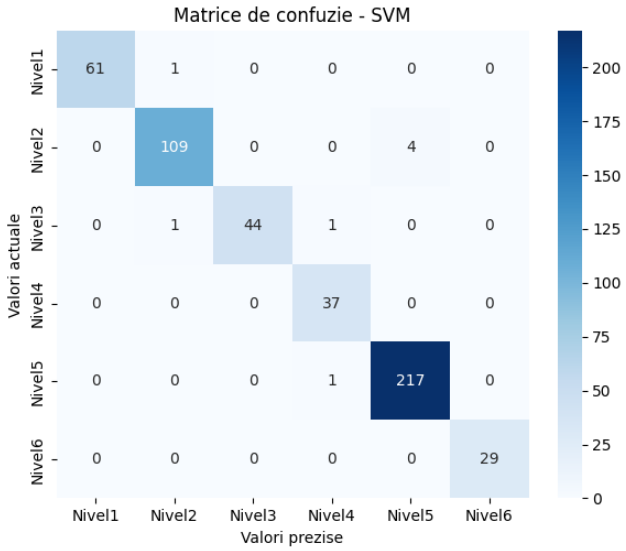

Figura 5.30. Matricea de confuzie pentru modelul SVM al gestului 3

Figura 5.30 prezintă matricea de confuzie pentru modelul SVM al gestului 3. Rezultatele cele mai bune, cu o precizie de predicție de 100% ale nivelelor le-a înregistrat nivelul 1, 3 și 6. Toate cele 134 de cazuri au fost prezise corect. La nivelul 2 și 4 al gestului 3, câte 2 cazuri au fost prezise greșit dintr-un total de 111, respectiv 39 cazuri de testare. Nivelul 5 a înregistrat și el performanțe bune, dintr-un total de 221 de cazuri de testare 4 au fost prezise greșit. Relația dintre numărul de teste și precizia de detecție a nivelului gestului nu se verifică. Chiar dacă s-a luat în considerare un număr scăzut sau ridicat de teste putem să înregistrăm rezultate bune pentru modelul SVM.

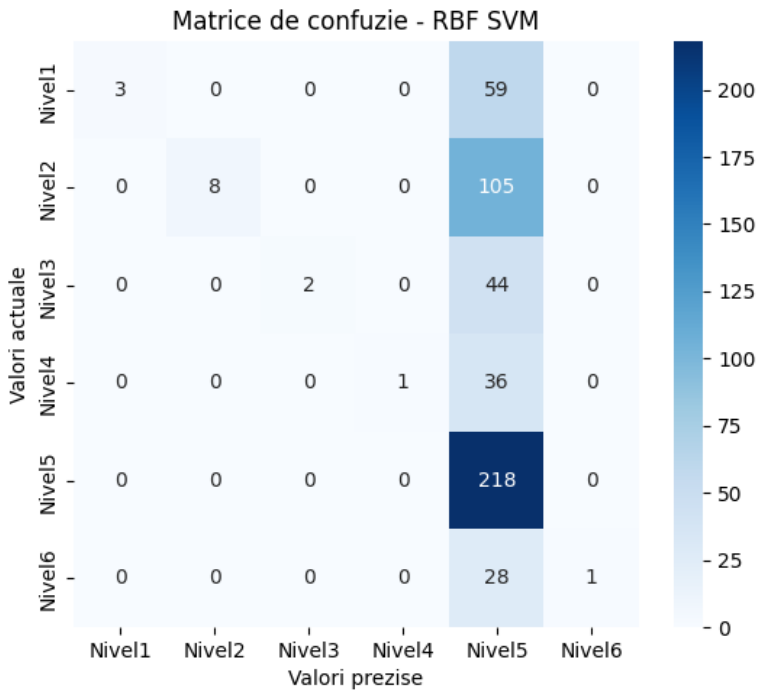

Figura 5.31. Matricea de confuzie pentru modelul RBF SVM al gestului 3

Din matricea de confuzie din figura 5.31 se observă că nivelul 5 a înregistrat cele mai bune rezultate. Toate cele 218 de cazuri de testare au fost prezise corect. În schimb celelalte niveluri au fost confundate cu nivelul 5. Nivelul 1 a fost prezis corect în 3 cazuri dint cele 62 de cazuri de testare, iar nivelul 2 a fost prezis corect în 8 cazuri dintr-un total de 113 cazuri de testare. Nivelurile 3, 4 și 6 la rândul lor au fost confundate cu nivelul 5. Pentru nivelul 3, doar 2 cazuri au fost prezise corect dintr-un toatal de 46 de cazuri de testare. Nivelurile 4 și 6 au doar un singur caz prezis corect fiecare, dintr-un total de 37, respectiv 29 cazuri de testare.

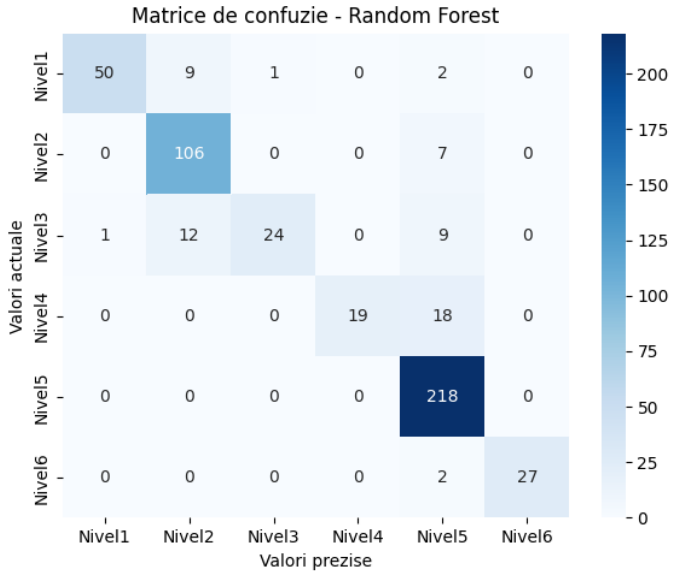

Figura 5.32. Matricea de confuzie pentru modelul Random Forest al gestului 3

Figura 5.32 prezintă matricea de confuzie pentru modelul Random Forest al gestului 3. Se observă că nivelul 5 a fost prezis cel mai bine. Acesta a înregistrat performanțe de 100% în prezicerea lui. Toate cele 218 cazuri de testare au fost prezise corect de modelul Random Forest pentru nivelul 5. Nivelul 6 a fost prezis și el bine. 27 de teste au fost prezise corect dintr-un total de 29 de cazuri de testare. Cu o precizie în detecție de aproximativ 50%, nivelul 4 a fost prezis corect în 19 cazuri dintr-un total de 37 de cazuri de testare. În 18 din acestea, nivelu 4 a fost confundat cu nivelul 5 al gestului. Nivelul 3 a fost prezis corect în 24 de cazuri dintr-un total de 46 cazuri de testare. O performanță bună pentru modelul Random Forest a fost înregistrată la nivelul 2 al gestului, doar 7 cazuri au fost prezise corect ca fiind nivelul 5 dintr-un total de 113 cazuri de testare. Nivelul 1 al gestului a fost prezis corect în 50 de cazuri dintr-un total de 62 de cazuri de testare. Acesta a fost confundat cu nivelul 2, 3 sau 5.

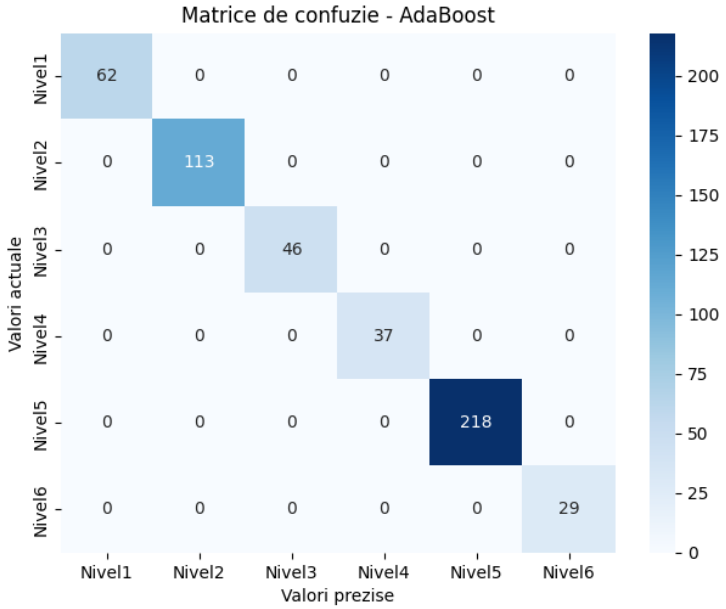

Figura 5.33. Matricea de confuzie pentru modelul AdaBoost al gestului 3

Matricea de confuzie pentru modelul AdaBoost al gestului 3 prezintă cele mai bune rezultate obținute. Astfel pentru toate cele 505 cazuri detecția nivelurilor este de 100%. Se poate observa că nu există nici o legătură între numărul de teste și performanța de detecție a nivelurilor gestului de flexie și extensie. La fel ca și modelul CART și modelul AdaBoost a înregistrat performanțele cele mai bune în clasificarea nivelurilor gestului de flexie și extensie.

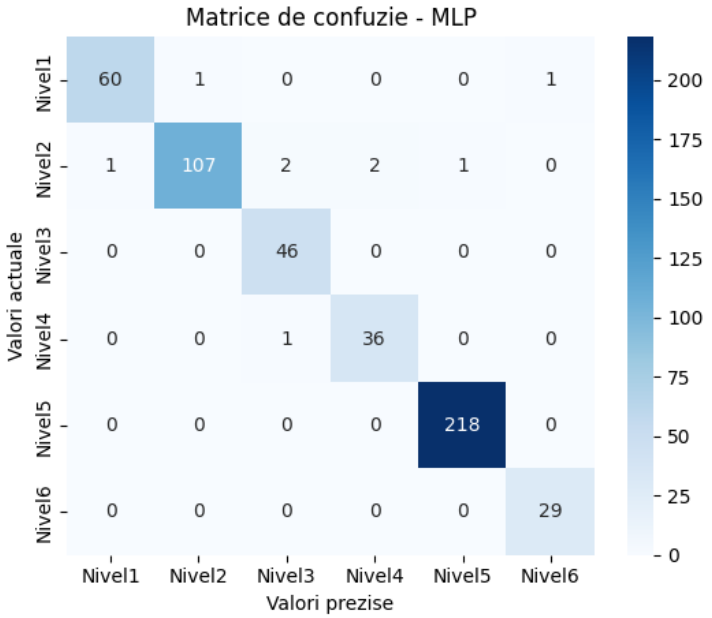

Figura 5.34. Matricea de confuzie pentru modelul MLP al gestului 3

Figura 5.34 prezintă matricea de confuzie pentru modelul MLP al gestului 3. Nivelurile 3, 5 și 6 au fost prezise în proporții de 100% cu acest model. Un total de 293 de cazuri de testare pentru cele 3 niveluri. Se observă lipsa de legătură dintre numărul de teste și precizia de prezicere a nivelurilor. De exemplu pentru nivelul 5 sunt 218 teste, iar pentru nivelul 6 sunt 29 de teste, toate fiind prezise corect. Nivelul 4 are un singur caz prezis greșit dintr-un total de 37 de cazuri, iar nivelul 1 are două cazuri prezise greșit dintr-un total de 62 de cazuri. 6 cazuri au fost prezise greșit la nivelul 2, fiind confundate cu nivelurile 1, 3, 4 sau 5 ale gestului. Un total de 113 cazuri de testare au fost folosite pentru nivelul 2 de către modelul MLP.

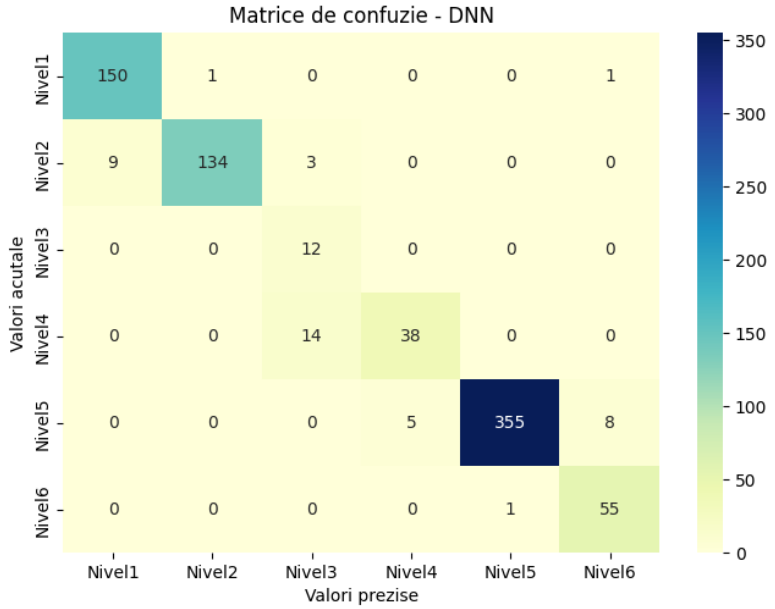

Figura 5.35. Matricea de confuzie pentru modelul DNN al gestului 3

Matricea de confuzie a modelului DNN este prezentată în figura 5.35. Nivelul 3 a fost prezis cu o precizie de 100% de către modelul DNN. Astfel toate cele 12 cazuri de testare au fost prezise corect. Pentru nivelul 1 și 6 modelul DNN a înregistrat performanțe bune, doar 2 cazuri pentru nivelul 1 au fost detectate greșit dintr-un total de 152 de cazuri de testare. Pentru nivelul 6 un singur caz din 56 de cazuri de testare a fost prezis greșit, fiind confundat cu nivelul 5. Nivelul 2, 4 și 5 au fost prezise cu o precizie mai scăzută. 12 cazuri au fost prezise greșit dintr-un total de 146 de cazuri de testare. Nivelul 4 a avut 52 de cazuri de testare, iar 14 dintre acestea au fost prezise greșit, fiind confundate cu nivelul 3. Pentru nivelul 5 s-au efectuat 368 de teste, iar 13 dintre acestea au fost greșite. S-a produs o confuzie cu nivelul 4 sau cu nivelul 5.

Pentru o vizualizare mai completă în tabelul 5.36 sunt afișate toate rezultatele obținute pe fiecare model în parte pentru fiecare dintre cele 3 gesturi. Astfel aici sunt afișate rezultatele pentru toate cazurile de test din matricele de confuzie. Am evidențiat cu culoarea albastră toate nivelurile gesturilor care au fost prezise cu o precizie de 100% de către cele 12 modele utilizate, iar cu culoarea roșu toate nivelurile gesturilor care nu au fost prezise, precizie de 0% de către cele 12 clasificatoare. Ca și notații în tabelul de mai jos am ales să utilizez "Nivel" și "nivel" pentru a face diferența între valorile actuale ale gesturilor și valorile prezise.

Model	Gest 1			Gest 2			Gest 3						Nivele Prezise
	Nivel 1	Nivel 2	Nivel 3	Nivel 4	Nivel 5	Nivel 6	Nivel 1	Nivel 2	Nivel 3	Nivel 4	Nivel 5	Nivel 6	
LR	743	9	0	14	5	10	59	2	0	0	0	1	nivel 1
	6	276	2	0	0	0	2	104	0	0	0	0	nivel 2
	0	3	527	0	0	0	0	1	38	3	0	0	nivel 3
	32	0	0	159	2	0	0	0	8	33	2	0	nivel 4
	2	0	0	2	25	0	0	6	0	1	215	0	nivel 5
	5	0	0	0	1	46	1	0	0	0	1	28	nivel 6
LDA	680	9	0	3	1	6	51	0	0	0	0	0	nivel 1
	2	263	0	0	0	0	8	108	1	0	4	1	nivel 2
	0	16	529	0	0	0	0	0	35	4	0	0	nivel 3
	62	0	0	170	1	0	0	0	10	31	0	0	nivel 4
	30	0	0	2	28	2	0	5	0	2	214	0	nivel 5
	14	0	0	0	3	48	3	0	0	0	0	28	nivel 6
KNN	779	1	0	3	4	3	60	1	0	0	0	0	nivel 1
	1	287	1	0	0	0	2	111	2	0	0	0	nivel 2
	0	0	528	0	0	0	0	0	43	5	0	0	nivel 3
	7	0	0	172	0	0	0	0	1	28	0	0	nivel 4
	0	0	0	0	29	0	0	1	0	4	218	1	nivel 5
	1	0	0	0	0	53	0	0	0	0	0	28	nivel 6
CART	786	0	0	0	0	0	62	0	0	0	0	0	nivel 1
	0	285	3	0	0	0	0	113	0	0	0	0	nivel 2
	0	3	526	0	0	0	0	0	46	0	0	0	nivel 3
	2	0	0	175	2	0	0	0	0	37	0	0	nivel 4
	0	0	0	0	30	0	0	0	0	0	218	0	nivel 5
	0	0	0	0	1	56	0	0	0	0	0	29	nivel 6
NB	578	5	0	1	3	1	60	1	0	0	0	0	nivel 1
	58	267	40	0	0	0	2	110	0	0	0	0	nivel 2
	0	16	489	0	0	0	0	2	46	0	0	0	nivel 3
	71	0	0	167	0	0	0	0	0	37	14	0	nivel 4
	22	0	0	0	30	0	0	0	0	0	199	0	nivel 5
	59	0	0	0	0	55	0	0	0	0	5	29	nivel 6
SVC	787	84	48	18	21	14	18	0	0	0	0	0	nivel 1
	0	204	0	0	0	0	0	20	0	0	0	0	nivel 2
	0	0	481	0	0	0	0	0	15	0	0	0	nivel 3
	1	0	0	157	0	0	0	0	0	4	0	0	nivel 4
	0	0	0	0	12	0	44	93	31	33	218	24	nivel 5
	0	0	0	0	0	42	0	0	0	0	0	5	nivel 6
Liniar SVM	784	0	0	1	3	1	61	0	0	0	0	0	nivel 1
	0	288	2	0	0	0	1	109	1	0	0	0	nivel 2
	0	0	527	0	0	0	0	0	44	0	0	0	nivel 3
	3	0	0	172	0	0	0	0	1	37	1	0	nivel 4
	0	0	0	0	30	0	0	4	0	0	217	0	nivel 5
	1	0	0	0	0	55	0	0	0	0	0	29	nivel 6
RBF SVM	788	162	252	108	23	35	3	0	0	0	0	0	nivel 1
	0	126	0	0	0	0	0	8	0	0	0	0	nivel 2
	0	0	277	0	0	0	0	0	2	0	0	0	nivel 3
	0	0	0	67	0	0	0	0	0	1	0	0	nivel 4
	0	0	0	0	10	0	59	105	44	36	218	28	nivel 5
	0	0	0	0	0	21	0	0	0	0	0	1	nivel 6
Random Forest	784	14	2	19	9	27	50	0	1	0	0	0	nivel 1
	4	272	1	0	0	0	9	106	12	0	0	0	nivel 2
	0	3	526	0	0	0	1	0	24	0	0	0	nivel 3
	0	0	0	156	2	0	0	0	0	19	0	0	nivel 4
	0	0	0	0	22	0	2	7	9	18	218	2	nivel 5
	0	0	0	0	0	29	0	0	0	0	0	27	nivel 6
AdaBoost	716	153	0	6	15	1	62	0	0	0	0	0	nivel 1
	0	134	529	0	0	0	0	113	0	0	0	0	nivel 2
	0	0	0	0	0	0	0	0	46	0	0	0	nivel 3
	70	0	0	168	0	37	0	0	0	37	0	0	nivel 4
	2	0	0	1	18	0	0	0	0	0	218	0	nivel 5
	0	0	0	0	0	18	0	0	0	0	0	29	nivel 6
MLP	787	5	0	28	5	13	60	1	0	0	0	0	nivel 1
	1	283	15	0	0	0	1	107	0	0	0	0	nivel 2
	0	0	514	0	0	0	0	2	46	1	0	0	nivel 3
	0	0	0	147	0	0	0	2	0	36	0	0	nivel 4
	0	0	0	0	28	0	0	1	0	0	218	0	nivel 5
	0	0	0	0	0	43	1	0	0	0	0	29	nivel 6
DNN	49	0	0	55	58	83	150	9	0	0	0	0	nivel 1
	2	54	0	0	0	0	1	134	0	0	0	0	nivel 2
	0	7	64	1	0	0	0	3	12	14	0	0	nivel 3
	0	0	0	0	0	0	0	0	0	38	5	0	nivel 4
	0	0	0	0	17	0	0	0	0	0	355	1	nivel 5
	0	0	0	0	0	0	1	0	0	0	8	55	nivel 6

Tabelul 5.36 Rezultatele modelelor pentru gesturile 1, 2 și 3

5.5. Concluzii

În acest capitol a fost prezentat cum pot fi clasificate 3 gesturi: gestul de strângere și deschidere al mâini, gestul de rotire al palmei și gestul de flexie și

extensie al mâinii, printr-o serie de 12 modele de clasificare folosite pe două rețele neuronale convoluționale. Varietatea mare de caracteristici ce descriu gesturile din setul de date permit construirea a noi gesturi care pot să aibă aplicabilitate în recuperarea mobilității mâinilor. Cele 29 de caracteristici ce descriu primele gesturi au adus un plus în clasificarea acestora, modelele de clasificare având o performanță mai ridicată după aplicarea caracteristicilor: vectorul direcție al mâinii și al degetului mare, unghiul de rotire al mâinii dar și semiperimetrul format de figura geometrică cu puncte la vârful degetului mare, mijlociu și mijlocul palmei. Nu în ultimul rând cele trei caracteristici: rotirea, înclinarea și grația din cele 14 caracteristici ce descriu gestul de flexie și extensie conduc la performanțe ridicate legate de clasificarea acestui gest.

Deoarece cele trei gesturi descrise au aplicabilitate în recuperarea mobilității mâinii toate gesturile au fost împărțite pe nivele de siguranță cu scop de monitorizare al progresului recuperării.

Principalele beneficii aduse în domeniul Calculatoare și Tehnologia Informației din acest capitol sunt: seturile de date variate ce pot fi folosite pe viitor în construirea de noi gesturi, dar și în stabilirea unor intervale de valori standard pentru definirea precisă a nivelului gestului. Un alt beneficiu îl reprezintă selectarea corectă a modelului de clasificare a unui gest dintr-o serie de 12 modele de clasificare.

După o analiză mai amănunțită a tabelelor 5.8 și 5.9 cu acuratețea modelelor se poate afirma că modelul CART (Clasificator de Arbore de Decizie) poate fi folosit pentru clasificarea gesturilor. Acest model se regăsește pe primul loc din punct de vedere al acurateței de clasificare, având o acuratețe de 99%. Chiar dacă sunt folosite două seturi de date diferite pentru primele două gesturi, respectiv ultimul, acest model prezintă cele mai bune performanțe. În plus am observat că pentru acest model nu există nici o relație între numărul de teste și precizia de clasificare a nivelului gesturilor. Chiar dacă sunt puține sau multe teste, acest model are aceeași precizie de clasificare.

Dacă analizăm tabelul 5.36 se observă ca modelele AdaBoost și DNN au avut rezultate diferite pe cele două seturi de date. Principalul motiv pentru care modelul AdaBoost a avut rezultate foarte bune pe al doilea set de date este legat de clasificatorii slabi pe care acesta se bazează. Deoarece modelul AdaBoost la sfârșitul antrenării combină clasificatorii slabi (caracteristicile gesturilor) pentru a produce un clasificator puternic, caracteristicile ce definesc gestul 3 sunt optime a fi luate în considerare. Deși pentru gesturile 1 și 2 sunt mai multe date în setul de date, dar și mai multe caracteristici, acestea nu pot fi luate în considerare pentru a le combina și a produce un clasificator puternic.

Modelul DNN a avut performanțe slabe (Tabelul 5.36) la primele două gesturi la fel ca și modelul AdaBoost deoarece acesta se bazează pe o rețea neuronală cu un număr de straturi și neuroni conectate între ele. Performanța slabă pentru gestul 1 și 2 a acestui model este dată în primul rând de numărul de neuroni, dar și de diversitatea caracteristicilor celor două gesturi. Modelele AdaBoost și DNN au avut performanțe slabe în primul rând datorită diversității caracteristicilor celor două gesturi, dar și a numărului de date de antrenare.

Diversitatea caracteristicilor gesturilor și numărul de date din setul de date este un plus pentru modelul CART. Se observă că acesta a avut performanțele cele mai bune. Acest lucru este datorat modului în care modelul este construit. Acesta construiește un arbore de decizie care se bazează în primul rând pe caracteristici,

fiecare nod din acest arbore reprezentând o caracteristică. Astfel un set de date cu numeroase caracteristici va duce la performanțe ridicate în ceea ce privește clasificare gesturilor.

Două modele care au avut performanțe diferite pe cele două seturi de date ale gesturilor le reprezintă DNN (Clasificator cu model de rețea neuronală cu învățare profundă) și AdaBoost. Acestea au avut acuratețea de 47% și 60% pentru primul set de date, iar pentru al doilea 95%, respectiv 99%. Se observă că acestea se află la poluri diferite din punct de vedere al acurateței în clasificarea gesturilor.

Afirmațiile de mai sus pot fi observate și în tabelul 5.35 care arată că modelul CART este cel mai bun pentru a fi utilizat în clasificarea gesturilor. La acest model precizia de clasificare pentru unele niveluri ale gesturilor s-a ridicat la 100%. Tot aici se observă și că modelele DNN și AdaBoost se află la poli opuși pentru fiecare set de date în parte.

Metricile prezentate în tabelele 5.9 și 5.10 întăresc afirmațiile de mai sus. Modul lor de calculare este explicat în capitolul 3, unde se regăsește și modul de construire al unui gest pentru dispozitivul Leap Motion.

Seturile de date au fost create prin utilizarea unor aplicații 3D create de mine care se bazează pe realitate virtuală. Aceste aplicații vor fi explicate și exemplificate în capitolul 6. Principalul scop al capitolului 6 fiind prezentarea utilității gesturilor în aplicații 3D bazate pe realitate virtuală, dar și cum poate fi monitorizată sau măsurată activitatea/interactivitatea într-o aplicație 3D.

6. GESTURILE ÎN APLICAȚII 3D BAZATE PE REALITATE VIRTUALĂ

În acest capitol voi prezenta cum se pot aplica gesturile recunoscute de dispozitivul Leap Motion în aplicații 3D bazate pe conceptul de Realitate Virtuală (VR). În plus, voi prezenta modul de interacțiune/control prin gesturi făcute cu mișcarea capului utilizatorului într-o aplicație VR.

Prin conceptul de aplicație de realitate virtuală se înțelege o aplicație creată de om pe calculator, care simulează realitatea atât de bine, astfel încât utilizatorii acesteia capătă impresia de prezență fizică. De multe ori realitatea virtuală este asociată cu un dispozitiv/ochelari de realitate virtuală sau căști pentru realitatea virtuală, dar aceasta poate fi asociată totodată cu o aplicație de simulare a realității. Elementul comun al acestui concept este reprezentat de partea 3D, care la rândul ei copiază realitatea. Practic orice aplicație care poate respecta acest concept poate fi asociată cu realitatea virtuală. Ochelari/căștile VR sunt doar dispozitive care au rol de a apropia utilizatorul de elementele 3D dezvoltate în aplicație.

Întrucât acest capitol este destinat prezentării unor serii de aplicații în care controlul se face prin gesturi LM sau gesturi făcute prin mișcarea capului, în cele ce urmează voi prezenta principalele puncte pe care doresc să le ating:

- Crearea unui caz de utilizare pentru folosirea unor gesturi LM în aplicații 3D – VR.
- Crearea unui caz de utilizare pentru folosirea unor gesturi VR în aplicații VR.
- Prezentarea unor aplicații educaționale cu folosirea gesturilor LM.
- Prezentarea unor aplicații educaționale cu folosirea gesturilor în VR.
- Prezentarea metricilor pentru măsurarea interacțiunii în aplicații 3D – VR.
- Prezentarea principalelor concluzii după aplicarea gesturilor în aplicații 3D.

6.1. Caz de utilizare gest LM în aplicații 3D

În cazul de utilizare din figura 6.1 este exemplificat cum este utilizat și procesat un gest care să fie recunoscut de dispozitivul Leap Motion. Inițial, utilizatorul plasează mâinile deasupra dispozitivului LM, iar dacă cel puțin una dintre mâini este prezentă, LM va recunoaște mâna și punctele aferente vârfurilor și joncțiunilor degetelor. Aceste două subsisteme de recunoaștere ale mâinilor și ale punctelor degetelor mâinii stau la baza creării de noi gesturi. O dată recunoscute mâinile și degetele acestora se trece mai departe la mișcarea mâinii de către utilizator, aceasta având ca urmare schimbarea permanentă a datelor referitoare la pozițiile mâinilor și ale degetelor în spațiul 3D. Mai departe se face o procesare a datelor rezultate mișcării mâinilor pentru a putea construi formule de calcul de distanțe și unghiuri cu scop de creare de gest dinamic nou. Un ultim pas în construirea gestului este reprezentat de verificarea datelor procesate. Dacă datele procesate sunt cuprinse în intervalul de

date optim pentru gestul ce se dorește a fi definit atunci se creează noul gest, iar dispozitivul LM va putea recunoaște acest gest.

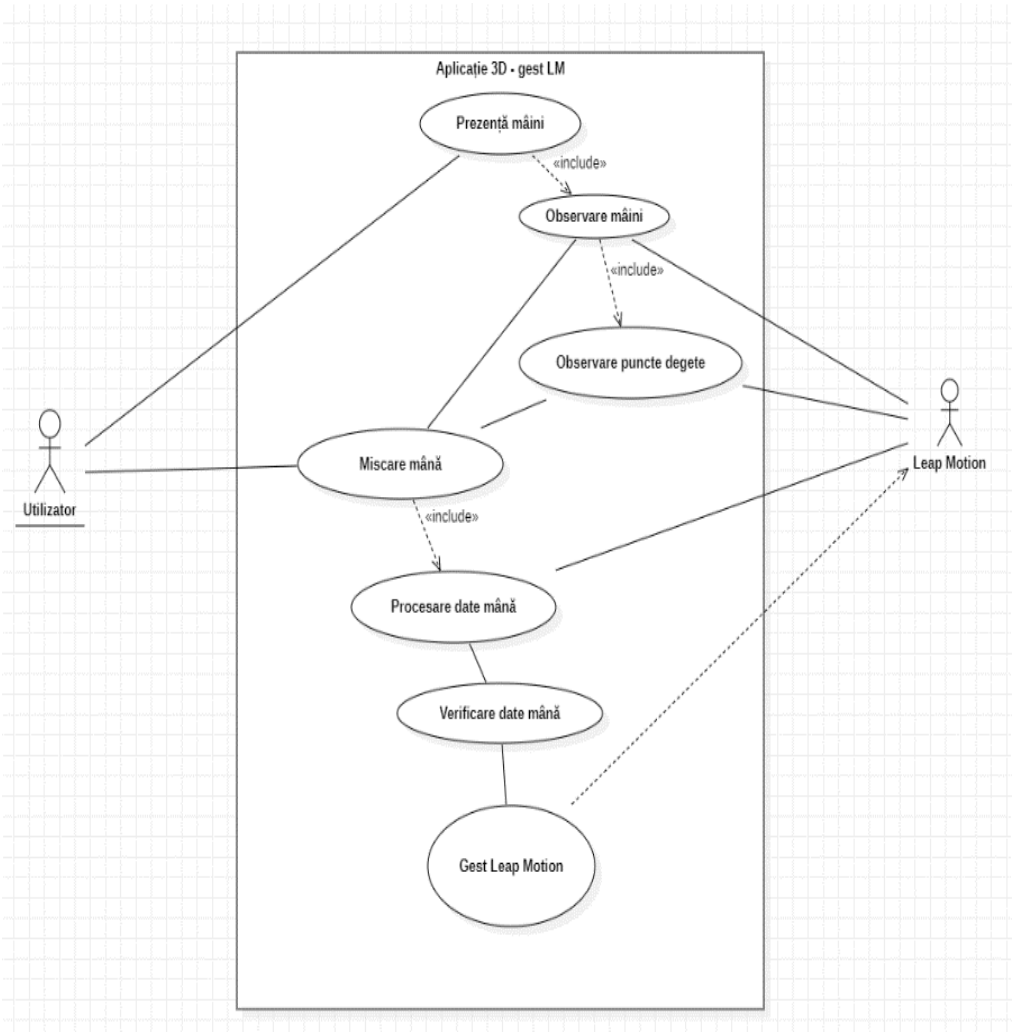

Figura 6.1. Caz de utilizare gest LM în aplicații 3D

6.2. Caz de utilizare gest LM în aplicații VR

Figura 6.2 arată cum se poate utiliza un gest făcut prin mișcarea capului în aplicații 3D bazate pe VR. Deoarece s-a dorit folosirea unui smartphone în care să fie prezentă o aplicație bazată pe VR, pentru început utilizatorul plasează telefonul în căștile VR. După plasarea telefonului în căști, acesta poate vizualiza mediul 3D al aplicației VR. Vizualizarea se face prin mișcarea capului sau a corpului utilizatorului

pe axa x sau y din spațiul 3D. Prin mișcarea capului sau a corpului utilizatorul observă mediul 3D, iar în plus va observa un selector cu ajutorul căruia acesta poate să interacționeze cu obiectele 3D. Pentru a putea interacționa se va procesa poziția selectorului și se va face o verificare a poziției acestuia față de obiectul dorit a fi selectat. Dacă acesta se găsește în aria de selecție a obiectului, atunci gestul de selecție este complet și se poate interacționa cu obiectul 3D.

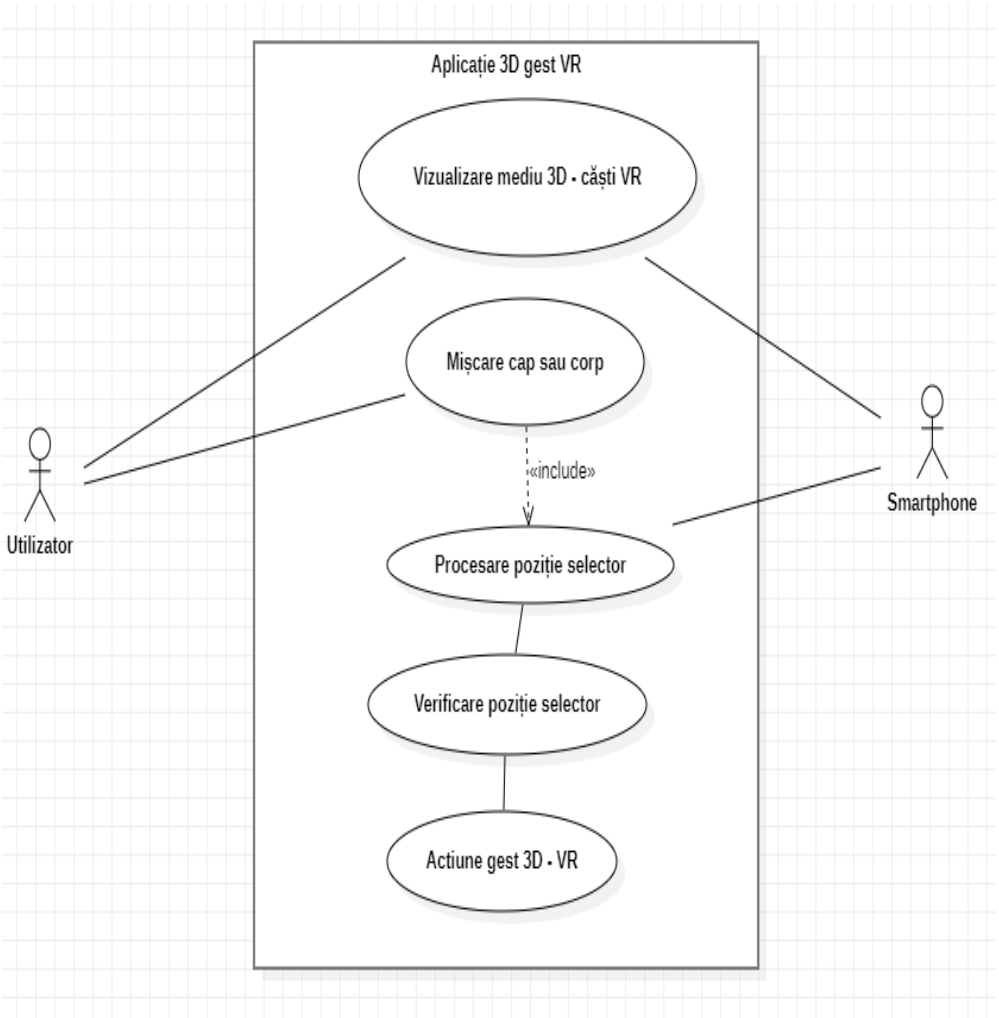

Figura 6.2. Caz de utilizare gest VR în aplicații 3D

6.3. Aplicații 3D cu aplicare a gesturilor LM

Aplicațiile 3D în care gesturile LM sunt utilizate aparțin domeniului educațional, dar și a domeniului medical, mai exact subdomeniul de recuperare medicală. În continuare voi prezenta pe scurt aplicațiile Skedu, AminoMotion și FlightLM. Gesturile prin care se face interacțiunea în aceste aplicații au fost discutate în capitolul 4 al acestei teze. În plus pentru a avea o vizualizare mai bună a acestora în spațiul 3D s-a utilizat 2 mâinii 3D virtuale din pachetul software LM care au rolul de a copia mișcarea mâinilor utilizatorului. Practic, aceste două mâinii se vor mișca la fel ca și mâinile reale ale utilizatorului.

6.3.1. Aplicația Skedu

Skedu este o aplicație 3D desktop cu ajutorul căreia utilizatorii pot învăța oasele sistemului osos. Interacțiunea în aceasta se face doar prin gesturi Leap Motion [NIC16]. Gesturile utilizate în această aplicație conduc fiecare la o acțiune/funcționalitate:

- Gestul de prindere – selectarea unui os din schelet.
- Gestul de glisare în sus și în jos mâna dreapta – rotirea obiectelor 3D pe axa X pozitiv sau negativ.
- Gestul de glisare în sus și în jos mâna stângă – mărirea și micșorarea obiectelor 3D
- Gestul de simulare a atingerii ecranului – schimbarea nivelelor/scenelor aplicației.

Prin gestul de prindere utilizatorii pot selecta/prinde oase din schelet și astfel află informații despre acestea (numele osului/grupului de oase). La prinderea acestora prin mâna virtuală, osul își schimbă culoarea. Această funcționalitate a fost dată pentru a observa mai bine selectarea. Gestul poate să fie executat fie prin folosirea degetului mare și arătător de la mână, fie prin folosirea degetului mare și a oricărui alt deget/degete de la mână. Este utilizat gestul complex de prindere, a cărui construire și funcționalitate este explicat în capitolul 4.

Gestul de glisare în sus și în jos al mâinii drepte este folosit pentru rotirea obiectelor. O glisare în jos a mâinii duce la o rotire a obiectului pe axa X cu un unghi predefinit, iar o glisare în sus a mâinii duce la rotirea obiectelor în celălalt sens pe axa X, la fel cu un unghi prestabilit. Practic acest gest simulează apropierea și depărtarea mâinii de dispozitivul LM.

Gestul de glisare în sus și în jos al mâinii stângi este folosit pentru mărirea și micșorarea obiectelor. La o mișcare de glisare a mâinii, obiectul se va mări (își va modifica dimensiunea/scala) cu o dimensiune stabilită, iar la o mișcare în jos a mâinii, acesta se va micșora cu aceiași dimensiune prestabilită.

Gestul de simulare a atingerii ecranului este asigurat prin mișcarea de apăsare virtuală a degetului arătător a unui ecran. Acest gest are rol în aplicația Skedu de a schimba scenele din aplicație.

Gesturile de glisare cât și gestul de apăsare a unui ecran sunt gesturi predefinite, pe care software-ul dispozitivului LM le poate detecta. Acestea sunt definite în biblioteca de gesturi predefinite prin numele: *Swipe* și *Screen_Tap*.

Pentru a vizualiza mai bine cum se poate utiliza această aplicație, se poate urmări următoarea adresă: https://www.youtube.com/watch?v=THueMLRohH8

În figura 6.3.1 se observă cum se face selecția/prinderea unui os din schelet. Aplicabilitatea acestei aplicații o constituie educația în domeniul medical.

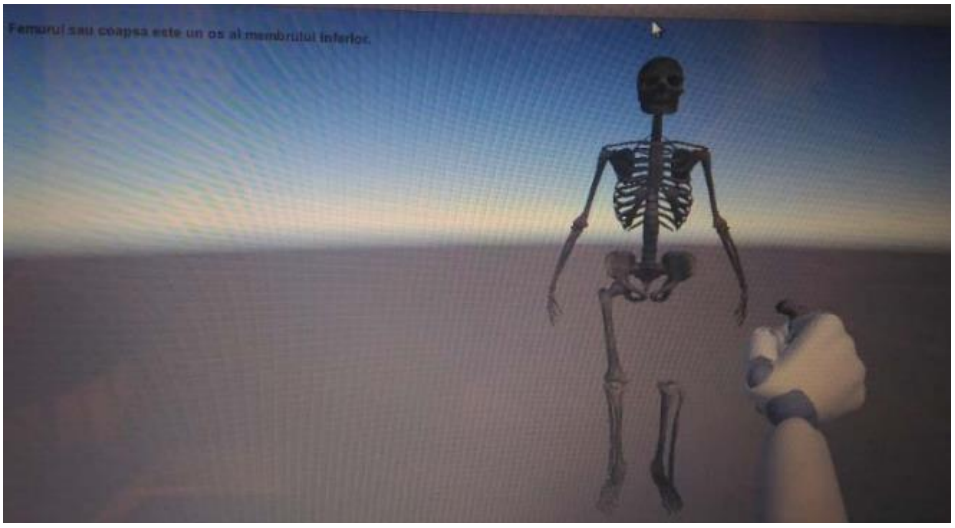

Figura 6.3.1. Skedu. Prinderea unui os din schelet

6.3.2. Aplicația AminoMotion

Această aplicație este destinată utilizatorilor care doresc să învețe aminoacizii dar și nucleotide ADN. Aplicația rulează pe desktop, este 3D și interacțiunea/controlul se face prin gesturi LM [1NIC17]. Ca și utilizare a acesteia, utilizatorii trebuie să selecteze o serie de trei nucleotide și să le combine într-un loc pentru a vizualiza rezultatul acestora. În aplicația 3D nucleotidele au fost considerate obiecte, cuburi de diferite culori. Fiecare culoare aplicată pe o serie de cuburi reprezintă o nucleotidă. Astfel cuburile albastre reprezintă guanina, cele roșii reprezintă citozina, cele verzi reprezintă adenina, iar cele galbene sunt asociate cu tinina sau uracilul. Pentru a interacționa cu acestea utilizatorul folosește gestul de prindere cu efect magnetic (algoritm explicat în capitolul 4 al tezei de doctorat). Acesta a fost utilizat pentru a crește rapiditatea și eficiența în utilizare pentru prinderea de obiecte. La apropierea degetului arătător de unul dintre cuburi acesta se lipește de deget, iar la plasarea degetului deasupra ariei de combinare a nucleotidelor (bolul de combinare al nucleotidelor) acestea vor cădea. La plasarea unei nucleotide în bolul de nucleotide utilizatorul va observa pe ecran numele acesteia. După ce s-a combinat 3 nucleotide

pe care utilizatorul le dorește, acesta va observa ce aminoacid a creat. În figura 6.3.2 se observă cum s-au combinat nucleotidele.

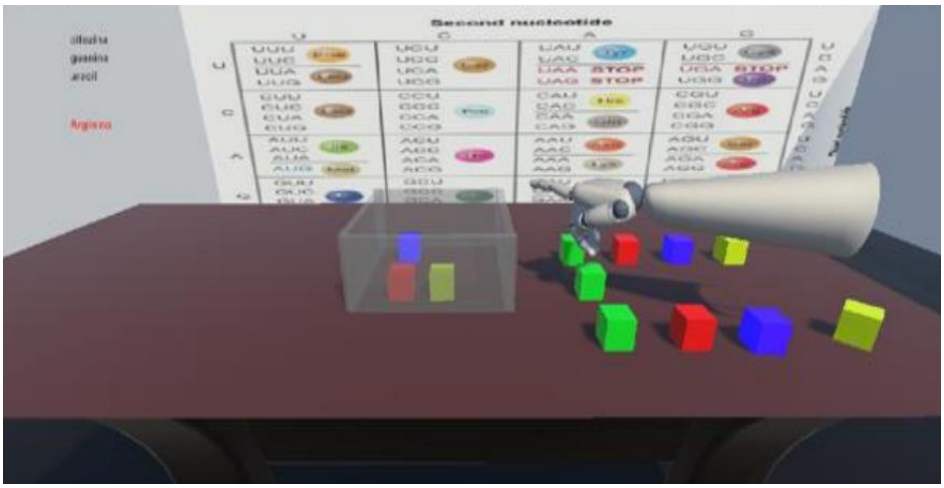

Figura 6.3.2. AminoMotion. Combinarea nuclotidelor – gest magnetic

6.3.3. Aplicația FlightLM

FlightLM este o aplicație 3D Desktop, în care utilizatorii pot controla un model 3D prin gestul de flexie și extensie al mâinii. Această aplicație este destinată utilizatorilor care doresc să experimenteze un alt mod de interacțiune între om și calculator față de cel clasic, mouse și tastatură. Nu în ultimul rând, controlul prin gestul de flexie și extensie poate fi utilizat în recuperarea mobilității articulației mâinii. Practic prin gestul de flexie și extensie, gest pe care oamenii îl reproduc pe parcursul recuperării, se poate utiliza în controlul unui model 3D. Algoritmul pentru gestul de flexie și extensie al mâinii este explicat în capitolul 4 al acestei teze. Acesta bazându-se pe calcularea unghiului format dintre mâna ce execută gestul și axa Y, înălțimea din scena 3D [NIC18].

În figura 6.3.3 se observă cum prim executarea unui gest de flexie obiectul 3D, avionul, este ridicat sau menținut pe o linie în care acesta poate să planeze. Gestul de extensie el mâinii face ca modelul 3D să coboare în spațiul 3D. Cele două gesturi, flexie și extensie, controlează avionul, fiecare fiind asociat cu ridicarea pentru flexie și cu coborârea pentru extensie. Scopul jocului FlightLM este de a controla un obiect 3D care se mișcă constant pe axa Z, astfel încât acesta să treacă prin obstacolele ce-i apar în cale. Generarea obstacolelor se face în mod aleator pe axa Y, astfel la rejucarea jocului utilizatorii nu vor experimenta același tipar de mișcare pentru gestul de flexie și extensie. În plus mișcarea pe axa Y a modelului 3D este direct proporțională cu mișcarea mâinii în gestul de flexie și extensie. O mișcare mai pronunțată, formarea unui unghi mai mare de 100 de grade între mână și axa Y, duce la o mișcare mai mare în sus sau în jos a obiectului.

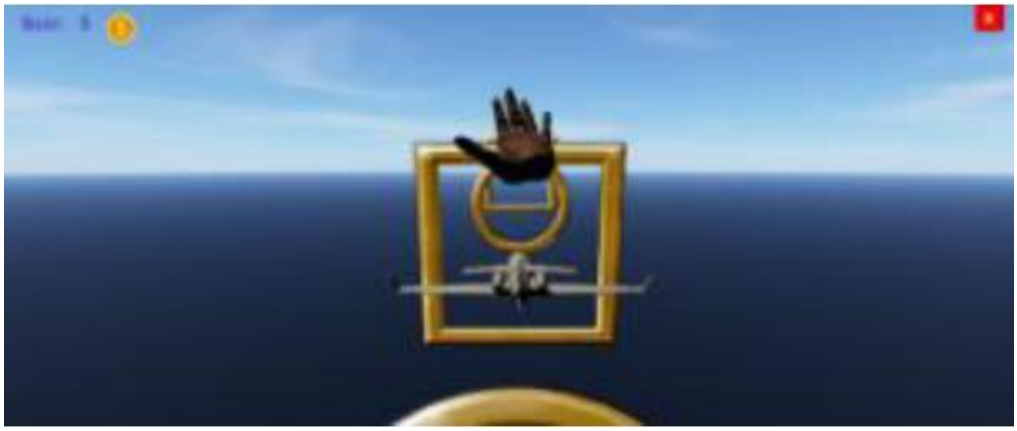

Figura 6.3.3. FlightLM. Controlarea unui obiect 3D prin gesturi LM

6.4. Gesturile în aplicații VR

Aplicațiile bazate pe realitate virtuală (VR) sunt destinate utilizatorilor ce doresc o apropiere și mai mare față de mediul VR. Controlul în astfel de aplicații se face, fie prin controlere/manete pe care utilizatorii le controlează cu scop de a interacționa cu mediul virtual, fie prin gesturi folosite cu același scop. Gesturile folosite pot fi făcute fie prin gesturi cu mâinile, fie prin gesturi cu mișcarea capului. În aplicațiile în care nu este posibil utilizarea gesturilor mâinilor, aplicații în care se utilizează un smartphone și căștile VR, cel mai comod și des utilizat gest este gestul făcut prin mișcarea capului. Practic prin mișcarea capului se poate interacționa cu obiectele 3D. Căștile VR sunt dispozitive în care se introduce telefonul mobil pe care rulează aplicația, iar acestea au rolul de a afișa aplicația 3D astfel încât să apropie utilizatorul de aplicație. Experimentarea într-un mod cât mai real al aplicației este dat de cele două lentile, plan convexe ale căștilor. În continuare voi prezenta două aplicații VR în care controlul se face prin gesturi făcute cu mișcarea capului: SkeduVR și BubbleVR.

6.4.1. Aplicația SkeduVR și BubbleVR

Cele două aplicații sunt destinate oamenilor care doresc să învețe într-un mod interactiv oasele scheletului uman, SkeduVR și respectiv algoritmul de sortare bubble sort, BubbleVR. Aceste aplicații se bazează pe pachetul software Google VR SDK, care permite construirea de aplicații 3D bazate pe realitate virtuală pentru telefoanele mobile împreună cu căștile VR. Cel mai mare plus al acestor tipuri de aplicații este dat de faptul că nu este nevoie de un buget mare al utilizatorului pentru a-și achiziționa

căștile VR. Costul unor căști VR decente nu depășește 100 de lei. Aplicațiile sunt educative pentru utilizatorii din domeniul medical sau din domeniul IT.

SkeduVR este o aplicație bazată pe realitate virtuală în care utilizatorii află informații despre fiecare os sau structură de oase din schelet. Modul de informare sau interacționare dintre utilizatori și scheletul 3D este realizat prin plasarea unui punct virtual pe suprafața oaselor. La plasarea punctului pe suprafața oaselor, utilizatorii observă selectarea osului și află informații despre acesta. Plasarea/mișcarea punctului în spațiul 3D se face prin mișcarea capului. La mișcarea capului (gestul de mișcare al capului), perspectiva utilizatorului se modifică. Astfel punctul poziționat tot timpul pe mijlocul ecranului telefonului mobil este adus deasupra suprafeței 3D a osului. Mișcarea în astfel de aplicații, prin gesturi făcute cu mișcarea capului, se face doar pe axele X și Y ale spațiului 3D. Alte funcționalități oferite în aplicația SkeduVR sunt mărirea și micșorarea modelului 3D, rotirea stânga și dreapta a modelului și resetarea scenei. Toate funcționalitățile realizându-se prin gestul de mișcare a capului. Plasarea punctului virtual pe unul dintre obiecte este asociată cu una dintre funcționalități [2NIC17]. În figura 6.4.1 este exemplificat cu la plasarea punctului virtual de culoare galbenă, pe oasele scheletului capului, utilizatorul primește informații despre acestea.

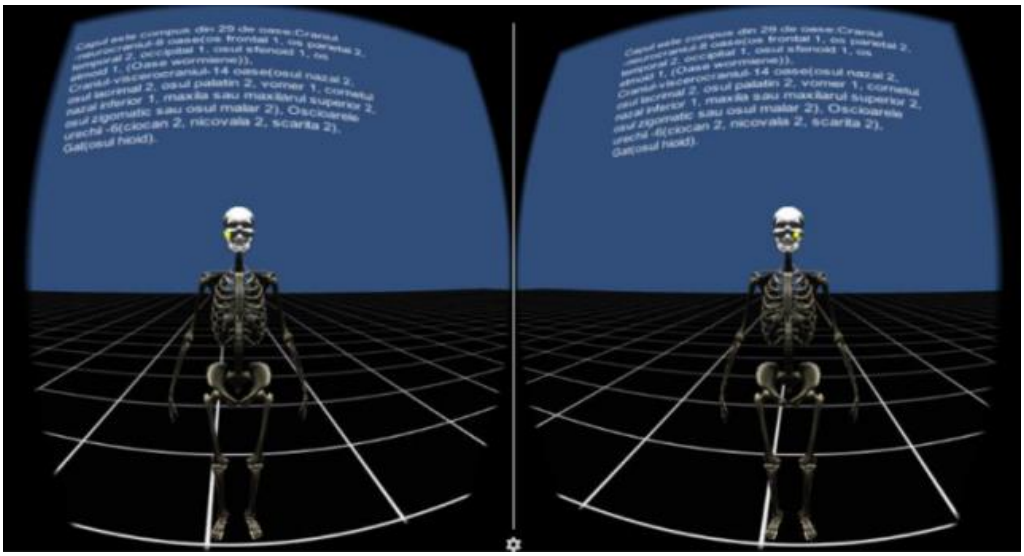

Figura 6.4.1. SkeduVR – Selectarea oaselor din schelet

BubbleVR este o aplicație VR, dezvoltată pentru cei care doresc să învețe algoritmul bubble sort. Aceasta se bazează pe conceptul de gamificare. Elementele care evidențiază gamificarea în această aplicație sunt: oferirea de sugestii utilizatorului în ceea ce privește sortarea, dar și cronometrarea timpului de sortare al elementelor. În aplicația BubbleVR controlul se face tot prin gesturi realizate prin mișcarea capului. Selectarea obiectelor/bilelor, se face prin plasarea punctului virtual pe suprafața acesteia. La selectarea bilei prin punctul virtual, aceasta primește poziția punctului și se lipește de acesta, până când este comparată cu bila vecină. Bilele 3D

din această aplicație, reprezintă elementele ce trebuie sortate. Acestea sunt de dimensiuni diferite, având fiecare înscripționat un număr. La început ele sun neordonate, iar utilizatorul trebuie să le ordoneze prin algoritmul bubble sort. Ordonarea se realizează prin selectarea bilei, iar la plasarea acesteia lângă bila vecină, din dreapta ei, se realizează compararea, iar dacă aceasta este mai mare atunci cele două bile se interschimbă. Acest proces se repetă până când toate bilele sunt sortate. Ca să întăresc și mai mult conceptul de gamificare implementat în BubbleVR, utilizatorii nu pot selecta orice bilă. Acestuia îi este permis să selecteze doar bila ce urmează a fi sortată.

În plus, pentru a vedea când bila a ajuns pe locul unde nu mai poate fi sortată, aceasta începe să se învârtă, evidențiind că sortarea ei s-a încheiat și urmează a fi selectată altă bilă. La sortarea completă, cronometrul ce pornește la începutul sortării, se oprește, iar utilizatorul primește feedback despre rapiditatea în executare a sortării. În aplicația BubbleVR, utilizatorii observă la începutul acestuia un tutorial exprimat printr-un video, în care pot învăța cum se interacționează cu obiectele 3D [2NIC18]. Figura 6.4.2 arată cum se văd bilele de către utilizator în spațiul 3D și cum trebuie să fie selectate.

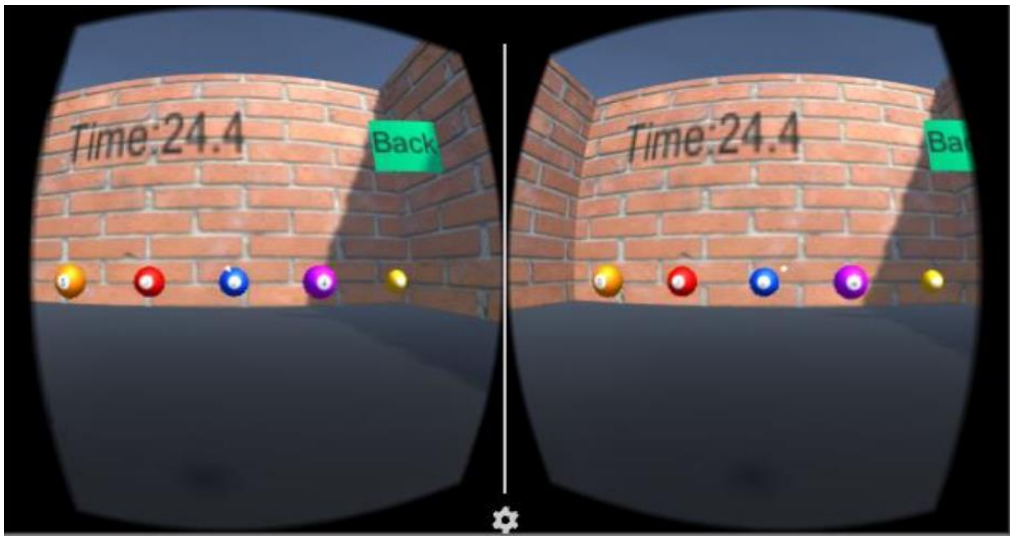

Figura 6.4.2. BubbleVR – Selectarea unui obiect pentru sortare

6.5. Evaluarea interactivității în aplicații 3D

În subcapitolele precedente am urmărit prezentarea unor aplicați educaționale, în care controlul se face prin gesturi. În continuare voi descrie cum se poate face detectarea interactivității în astfel de aplicații. De cele mai multe ori evaluarea interactivității aplicațiilor se realizează în mod clasic, în care utilizatorii/testorii aplicației primesc în prealabil un tutorial al utilizării aplicației, apoi

aceștia utilizează aplicația. La sfârșitul utilizării oamenii primesc un chestionar sau o serie de chestionare în care sunt întrebați despre aplicație. Toți acești pași se fac respectând un plan de testare.

Utilizatorii evaluează aplicația, iar în multe dintre întrebări sunt puși să răspundă "Cât de interactiv este un obiect/parte a aplicației/aplicația?" sau "Cât de ușor/greu au putut să execute un anumit task?". De cele mai multe ori la răspunsul unor astfel de întrebări apare subiectivismul din partea utilizatorilor. Aceștia fie nu au găsit funcționalitatea de care sunt întrebați, fie doresc să completeze cât mai rapid acel chestionar nemaifiind atenți la ceea ce răspund. Răspunsurile acestora fiind luate în considerare, chiar dacă acestea nu sunt reale.

Pentru a elimina acest inconvenient în testarea interactivității aplicațiilor propuse, am implementat un sistem de măsurare automată al acesteia. Deoarece toate aplicațiile implementate sunt realizate în editorul Unity, am folosit serviciul *Unity Analytics*, pentru a măsura interactivitatea. Prin acest serviciu se pot crea metrici personalizate, cu scop de măsurare a unor proprietăți/caracteristici pe care dezvoltatorul dorește să le urmărească. Printre metricile personalizate introduse de mine amintesc: timp de utilizare al aplicație/timp de interacțiune, prezența selectării sau numărul de utilizatori [1NIC20].

Forma unui eveniment de tipul *Unity Analytics* este exemplificat în scriptul C# din figura 6.5. Variabila *a* este inițializată cu metoda ***CustomEvent()*** care primește doi parametrii. Primul parametru face referire la numele evenimentului personalizat, în acest caz *Interactiune*, iar al doilea este un dicționar ce va conține numele obiectului cu care se interacționează, timpul de utilizare și scorul obținut. Această metodă poate fi chemată oricând în aplicație. De exemplu la apăsarea unui buton sau la selectarea unui obiect. Partea interesantă a unor astfel de evenimente de tipul *Analytics* este reprezentată de complexitatea cu care acestea pot fi create. În plus acestea pot fi observate în panoul de comandă al *Unity Analytics*. Pentru a folosii acest serviciu este important ca dezvoltatorii să aibă un cont *Unity* și să activeze serviciul *Unity Analytics* din editorul *Unity*.

```
var a = Analytics.CustomEvent("Interactiune", new Dictionary<string, object>
{
    { "Nume obiect", obiect.name},
    { "Timp", t },
    { "Scor", s }
});
```

Figura 6.5. Exemplu eveniment personalizat

6.6. Concluzii

În acest capitol am prezentat cum pot fi aplicate gesturile în aplicații 3D bazate pe realitate virtuală și cum se interacționează prin acestea cu obiectele 3D. Tot aici am arătat cum se utilizează gesturile făcute cu mâinile și gesturile făcute prin mișcarea capului.

Principalele beneficii aduse în domeniul Calculatoare și Tehnologia Informației sunt reprezentate de: utilizarea și procesarea gesturilor pentru interacțiunea în aplicațiile 3D bazate pe VR și măsurarea interactivității în aplicații 3D prin servicii Unity Analitycs.

Folosirea gesturilor LM și observarea mișcării mâinii în aplicații 3D desktop, prin mâinile virtuale, dar și folosirea gesturilor prin mișcarea capului în aplicații 3D smartphone, prin căștile VR, aduce la o apropiere între utilizator și calculator/smartphone. Se poate spune că prin utilizarea acestor tipuri de gesturi în spații 3D se respectă principiul realității virtuale care spune că utilizatorul trebuie să simtă o prezență fizică în acestea. Toate aceste caracteristici și principii stau la baza realității mixte (MR), care combină lumea virtuală cu cea reală pentru a apropia utilizatorul de aplicație. În plus aceasta folosește dispozitive de intrare (manete, senzori) pentru a facilita controlul aplicației. În lucrarea "Mixed Reality Supporting Modern Medical Education" se prezintă mai amănunțit ce înseamnă acest concept [1NIC18]. Totodată în ultimii ani dacă se face referire la astfel de tehnologii ca: realitate virtuală, augumentată sau mixtă, toate acestea se pot îngloba prin folosirea conceptului realitate extinsă sau experimentală (XR).

Chiar dacă s-a propus un nou mod de evaluare a interactivității în aplicații 3D, acesta nu vine cu scop de eliminare a evaluării clasice. Serviciile oferite de *Unity Analytics* sunt binevenite în a fi folosite pentru a valida unele cazuri din evaluarea clasică, prin eliminarea răspunsurilor fals pozitive/negative din chestionare în ceea ce privește interactivitatea.

7. Concluzii și direcții de cercetare

În acest capitol voi prezenta principalele concluzii care reies din teza de doctorat. Totodată voi arăta care sunt principalele direcții de cercetare și dezvoltare ale acestei teme.

7.1. Concluzii

Teza de doctorat cu tema Contribuții la interacțiunea multimodală în 3D cuprinde rezultatul cercetărilor efectuate în domeniul interacțiunii prin gesturi în aplicații 3D, utilizând algoritmi de IA care conduc la soluții precise în interpretarea mișcărilor palmei.

Principalul scop al cercetării este reprezentat de dezvoltarea de noi algoritmi ce definesc gesturi pentru dispozitivul Leap Motion. Gesturile definite pentru acest dispozitiv sunt: 3 gesturi de prindere, gestul de flexie și extensie a încheieturii mâinii, gestul de rotire a palmei, gestul de strângere și deschidere a mâinii, gestul complex de apropiere și depărtare al mâinilor. Totodată s-a urmărit creșterea preciziei de recunoaștere a gesturilor definite. Au fost utilizate tehnici de machine learning pentru a clasifica gesturile definite prin algoritmi. Modelele utilizate în clasificarea gesturilor definite sporesc precizia de clasificare a acestora. Principalele gesturi definite au aplicabilitate în domeniul educațional și în domeniul medical. Gesturile au aplicabilitate în domeniul educațional, deoarece prin utilizarea lor se intensifică interacțiunea dintre om și calculator în medii virtuale 3D. Pe de altă parte gesturile definite se pot utiliza în medicină, în special în ramura medicinei care se ocupă cu recuperarea mobilității mâinilor, deoarece acestea implică utilizarea gesturilor în procesul de recuperare.

În această teză de doctorat se arată cum poate fi utilizată realitatea virtuală pentru a crește interacțiunea om-calculator. Contribuțiile originale din această lucrare sunt reprezentate de definirea gesturilor cu ajutorul cărora se poate interacționa în aplicații bazate pe VR și definirea metricilor cu care se poate măsura interactivitatea în aplicații 3D.

În continuare voi puncta care sunt contribuțiile aduse care reies din fiecare capitol în parte ale acestei teze de doctorat.

Capitolul 1 de introducere prezintă:

Tema de doctorat aleasă și contextul acesteia, prezentându-se care a fost motivația alegerii acestei teme de doctorat: "Contribuții la interacțiunea multimodală în 3D";

Obiectivele principale ale tezei

- Crearea de algoritmi pentru detectarea gesturilor dinamice ale dispozitivului Leap Motion.
- Clasificarea gesturilor construite și creșterea preciziei de recunoaștere a acestora.

- Utilizarea tehnologiilor de realitate virtuală pentru gesturile construite. Măsurarea interactivității în aplicații 3D.

Împărțirea pe capitole, precum și relația dintre capitole (legătura capitolelor tezei de doctorat) este discutată în partea de introducere.

Capitolul 2 cuprinde o analiză a principalelor preocupări din literatura de specialitate în direcția interacțiunii prin gesturi și folosirea tehnologiilor de realitate virtuală și este structurat 3 părți:

În prima parte se realizează analiza stadiului actual în ceea ce privește definirea gesturilor, structurarea acestora, precum și clasificarea lor prin diferite metode. O primă metodă și cea mai cunoscută face referire la folosirea de formule matematice pentru definirea de gesturi. Altă metodă face referire la utilizarea rețelelor neuronale pentru clasificarea gesturilor. Dispozitivele cele mai cunoscute care sunt capabile să recunoască gesturi sunt prezentate în această parte a capitolului 2. Totodată se face o comparație între acestea din punctul de vedere al utilizării lor în domeniul medical, pentru recuperare. În plus se oferă o perspectivă asupra modelelor utilizate în clasificarea gesturilor pe bazele de date deschise ce le definesc. Sunt prezentate principalele modele cu metricele de performanță obținute. Deoarece teza de doctorat se concentrează pe definirea de noi gesturi pentru dispozitivul Leap Motion, majoritatea cercetărilor studiate fac referire la acest dispozitiv. Principalele gesturi găsite în literatura de specialitate fac referire la: gesturile alfanumerice, gesturile utilizate în educație și gesturile cu utilitate în recuperarea mobilității mâinilor.

A doua parte a acestui capitol este dedicată tehnologiei, și anume realitatea virtuală. Aici se face o prezentare a principalelor ramuri în care se aplică VR. Sunt definite și celelalte tehnologii care au fost create după apariția realității virtuale: realitatea augmentată (AR), realitatea mixtă (MR) și realitatea extinsă/experimentală (XR).

Ultima parte a acestui capitol face referire la îmbinarea primelor părți în construirea de aplicații bazate pe realitate virtuală în care interacțiunea se realizează prin gesturi. Se face o analiză a modului în care dispozitivele de recunoaștere a gesturilor sunt utilizate în aplicații 3D precum și a modului de măsurare a interactivității în astfel de aplicații.

Capitolul 3 prezintă bazele teoretice care stau la baza definirii și clasificării gesturilor recunoscute de dispozitivul Leap motion. Astfel am realizat următoarele:

Pașii principali ce trebuie făcuți pentru definirea unui gest recunoscut de dispozitivul Leap Motion. Aceștia fac referire la verificarea prezenței unui dispozitiv LM, a prezenței mâinilor în frameurile LM, procesarea pozițiilor mâinii și a degetelor, calculul de distanțe, unghiuri de rotire, direcții de vectori și suprafețe și verificarea valorilor calculate pentru inițializarea gestului.

Modelele de clasificare a gesturilor sunt prezentate în acest capitol. Fiecare model este prezentat atât din punctul de vedere teoretic, cât și din punctul de vedere experimental/practic pentru a facilita înțelegerea modului în care acestea funcționează.

Termenii și formulele utilizate în analiza statistică a datelor sunt prezentate la finalul acestui capitol. La fel ca și la modelele de clasificare prezentate anterior și formulele din statistică sunt însoțite de exemple.

Ultimele puncte ale acestui capitol fac referire la principalele beneficii aduse în domeniul Calculatoare și Tehnologia Informației. Acestea sunt date de folosirea de exemple în definirea unor termeni, modele de clasificare, și formule statistice, care să ajute utilizatorul să le înțeleagă pe deplin.

În **Capitolul 4** sunt prezentate primele contribuții aduse de teza de doctorat. În acest capitol se prezintă 7 algoritmi originali pentru 7 gesturi pe care dispozitivul Leap Motion le va recunoaște. Astfel în acest capitol se prezintă:

3 algoritmi pentru gestul de prindere: unul care implică utilizarea degetului mare și a degetului arătător, în altul se utilizează doar degetul arătător și în altul se utilizează degetul arătător și oricare dintre alte degete ale aceleiași mâini. Acest algoritm este definit prin pașii explicați la capitolul 3, iar formulele matematice principale folosite fac referire la distanța dintre două puncte în spațiul 3D.

Un algoritm face referire la gestul de flexie și extensie a încheieturii mâinii. La fel ca la ceilalți algoritmi se utilizează pașii definiți în capitolul 3, iar mai departe se procesează unghiul de înclinare al mâinii. Astfel s-au definit 3 intervale de valori pentru acest unghi. Acest lucru face ca acest gest să poată fi folosit la exercițiile de recuperare ale mobilității mâinilor

Un alt algoritm definit este algoritmul pentru gestul de rotire al palmei. La fel și la algoritmul precedent se ține cont de procesarea pașilor definiți în capitolul 3, iar mai departe la procesarea unghiului de rotire al palmei. Pentru recunoașterea cât mai bună a gestului de rotire de către LM, la acest algoritm s-a introdus și direcția de rotire a mâinii, precum și direcția degetului mare. Astfel s-au eliminat cazurile în care de făcea o detectare greșită a utilizării mâinii drepte sau stângi. Gestul creat prin algoritmul de detecție a gestului de rotire, poate face parte și el în rândul gesturilor cu utilitate în recuperarea mobilității mâinilor.

Un ultim gest cu utilitate în recuperarea mobilității mâinilor este gestul definit prin algoritmul de detecție a gestului de strângere și deschidere al mâinii. Algoritmul se bazează și el pe pașii definiți în capitolul 3, iar în plus se ține cont pentru detectarea corectă de semiperimetrul format de figura geometrică cu puncte la vârful degetului mare, mijlociu și mijlocul palmei. Astfel deschiderea și închiderea mâinii formează o figură geometrică, a cărui semiperimetru este direct proporțional cu închiderea și deschiderea mâinii. Întrucât gestul poate fi folosit în recuperare, acesta a fost împărțit pe nivele. Aceste nivele sunt definite de semiperimetru, iar valoarea lui pune la o solicitare mai mare sau mai mică a încheieturii degetelor mâinii.

Gestul complex definit prin algoritmul de detecție al gestului complex face referire la mișcarea de apropiere și depărtare a mâinilor. Deoarece marea majoritate a gesturilor definite prin algoritmi pentru dispozitivul LM fac referire la utilizarea unei singure mâini, acesta se bazează pe utilizarea ambelor mâini. Astfel la algoritmul acestui gest se fac procesări legate de distanța dintre mâini. Mai precis se calculează distanța dintre mijlocul palmelor. Dacă ambele mâini sunt prezente în frame și distanța este diferită de zero, atunci are loc gestul complex.

În acest capitol sunt prezentați algoritmi de detectare a gesturilor LM. Aceste gesturi având utilitate atât în domeniul educațional, cât și în domeniu medical.

Principalele beneficii aduse în domeniul Calculatoare și Tehnologia Informației sunt date de cei 7 algoritmi definiți pentru detectarea gesturilor LM.

Capitolul 5 reprezintă al doilea capitol cu contribuții aduse în Interacțiunea multimodală în 3D din domeniul Calculatoare și Tehnologia Informației. Acest capitol prezintă contribuțiile aduse la precizia și acuratețea gesturilor dinamice pentru dispozitivul LM. Principalii pași pe care i-am urmat pentru a crește precizia de clasificare a gesturilor LM:

Colectarea datelor de la dispozitivul LM. Acestea fac referire la poziții ale vârfului degetelor în spațiul 3D, poziții ale mijlocului palmei, distanțe dintre degetul mare și celelalte degete de la mână, distanțe dintre vârful degetelor și mijlocul palmei, vector direcție al degetului mare de la mână, unghiul de rotire al mâinii. Au fost colectate 2 seturi de date ce descriu 3 gesturi definite în capitolul 4. Setul de date face referire la gesturile ce pot fi utilizate în recuperare.

Filtrarea datelor în funcție de numărul de date colectate, maxim 10 valori identice pentru fiecare caracteristică. Pentru filtrare s-au utilizat intervale de valori ce definesc gesturile descrise de datele colectate.

Selectarea modelului de rețea neuronală a fost făcută după aplicarea a 12 modele. Toate cele 12 modele au fost definite și exemplificate în capitolul 3 al tezei de doctorat.

După aplicarea modelelor pe seturile de date s-au obținut performanțe diferite. Astfel rezultatele obținute a fiecărui model au fost exemplificate prin matrice de confuzie. În plus au fost prezentate metricele de performanță: precizia, recall, f1 scor și acuratețea.

Analiza rezultatelor obținute a dus la concluzionarea că modelul CART este cel mai optim model care poate fi folosit. Acest model se găsește pe primul loc din punct de vedere a acurateței obținute: 99%. În plus s-a observat că acest model nu ține cont de numărul de date de testare. El înregistrând performanțe ridicate atât pe date puține cât și pe date multe de testare.

Tot prin analiza rezultatelor obținute de cele 12 modele aplicate se observă care sunt modelele cu rezultate opuse pe cele două seturi de date. Cele două modele cu care au înregistrat astfel de performanțe sunt DNN și AdaBoost.

Capitolul 6 prezintă o serie de aplicații în care au fost utilizate gesturile LM. În plus sunt prezentate diferite perspective din punctul de vedere al utilizării realității virtuale. Principalele contribuții aduse prin acest capitol sunt:

Crearea și prezentarea unui caz de utilizare pentru folosirea gesturilor LM definite în celelalte capitole în aplicații VR.

Prezentarea unor aplicații educaționale care folosesc pentru interacțiune gesturi LM și/sau gesturi pentru căștile VR.

Utilizarea serviciilor oferite de platforma Unity Analytics și definirea metricilor pentru măsurarea interactivității în aplicații 3D.

7.2. Lista de publicații

Prin rezumarea contribuțiilor aduse la teza de doctorat "Contribuții la interacțiunea multimodală în 3D", în anii de studiu am realizat 19 lucrări de cercetare care sunt indexate ISI/WOS sau care urmează să se indexeze, acestea fiind indexate BDI. În continuare voi prezenta lista de publicații:

1. **Nicola, S**; Stoicu-Tivadar, L; Virag, I; Crisan-Vida, M, "Leap motion supporting medical education", in Proc. 12th IEEE International Symposium on Electronics and Telecommunications (ISETC 2016), pp. 153-156, Timișoara, România, WOS: 000390717800035 (ISI)

2. **Nicola, S**; Virag, I; Stoicu-Tivadar, L, "VR Medical Gamification for Training and Education", in Proc. 11th Annaul Conference on Health Informatics Meets eHealth (eHealth 2017), Vol. 236, pp. 97- 103, Schloss Schonbrunn, Austria, WOS: 000426828000013 (ISI)

3. **Nicola, S**; Handrea, FL; Crisan-Vida, M; Stoicu-Tivadar, L, "DNA Encoding Training Using 3D Gesture Interaction", in Proc. Special Topic Conference of the European-Fedeeration-for-Medical-Informatics (EFMI STC 2017), Vol. 244, pp. 63-67, Tel Aviv, Israel, WOS: 000450270500013 (ISI)

4. **Nicola, S**; Stoicu-Tivadar, L, "Hand Rehabilitation Using a 3D Environment and Leap Motion Device", Studies in health technology and informatics (ICIMTH 2018), Vol. 251, pp. 43-46, ISSN: 0926-9630

5. Mahmut, E-E; **Nicola, S**; Stoicu-Tivadar, V, "A Computer-Based Speech Sound Disorder Screening System Architecture", Studies in health technology and informatics (ICIMTH 2018), Vol. 251, pp. 39-42, ISSN:1879-8365

6. **Nicola, S**; Lupșe, OS, Stoicu-Tivadar, L, "Novel Gesture Interaction Using Leap Motion in 3D Applications", in Proc. 12th International Symposium on Applied Computational Intelligence and Informatics (SACI 2018), Timișoara, Romania, May 2018, pp. 113-118, WOS:000448144200020 (ISI)

7. **Nicola, S**; Stoicu-Tivadar, L, "Mixed Reality Supporting Modern Medical Education", in Proc. Special Topic Conference of the European-Fedeeration-for-Medical-Informatics (EFMI STC 2018), Vol. 255, pp. 242-246, Zagreb, Croatia, WOS: 000455957400047 (ISI)

8. **Nicola, S**; Stoicu-Tivadar, L; Patrascoiu, A, " VR for Education in Information and Tehnology: application for Bubble Sort", in Proc. 13th International Symposium on Electronics and Telecommunications (ISETC 2018), pp. 343-346, Timișoara, România, WOS: 000463031500076 (ISI)

9. **Nicola, S**; Chirila, OS.; Stoicu-Tivadar, L, "Enhancing Precision in Gesture Detection for Hand Recovery fter Injury Using Leap Motion and Machine Learning", in Proc. 18th International Conference on Informatics, Management and Technology in

Healthcare (ICIMTH 2019), Vol. 262, pp. 320-323, Athens, Greece, WOS:000560388600073 (ISI)

10. Mahmut, E-E; **Nicola, S**; Stoicu-Tivadar, V, "CROSS-CORRELATION BASED AUTOMATIC SEGMENTATION OF MEDIAL PHONEMES", in Proc. 14th International Symposium on Electronics and Telecommunications (ISETC 2020), pp. 293-296, Timișoara, România, WOS: 000612681000070 (ISI)

11. Mahmut, E-E; **Nicola, S**; Stoicu-Tivadar, V, "Cross-Correlation Based Automated Segmentation of Audio Samples", in Proc. 18th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2020), Vol. 272, pp. 241-244, Athens, Greece, WOS:000630065600062 (ISI)

12. **Nicola, S**; Stoicu-Tivadar, L "Evaluating Interactivity in VR Healthcare Applications Using Analytics", in Proc. 18th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2020), Vol. 272, pp. 225-228, Athens, Greece, WOS:000560388600073 (ISI)

13. **Nicola, S**; Chirila, OS; Stoicu-Tivadar, L, "Gesture Classification for a Hand Controller Device Using Neural Networks", 30th Medical Informatics Europe (MIE 2020) Conference, Vol. 270, pp. 756-760, APR, 2020, WOS:000630065600058 (ISI)

14. Mahmut, E-E; **Nicola, S**; Stoicu-Tivadar, V, "Word-Final Phoneme Segmentation Using Cross-Correlation", Studies in health technology and informatics (STC 2020), Vol. 275, pp. 132-136, Nov, 2020, ISSN:1879-8365

15. Varga, G; Stoicu-Tivadar, L; **Nicola, S**, "Serious Gaming and AI Supporting Treatment in Rheumatoid Arthritis", 31th Medical Informatics Europe (MIE 2021) Conference, Vol. 281, pp. 699-703, Mai, 2021.

16. **Nicola, S**; Stoicu-Tivadar, L, "E Sharing the IT Educational Experience of Developing 3D Applications for Medical Students Training", 19th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2021), Vol. 289, pp. 204-207.

17. Mahmut, E-E; **Nicola, S**; Stoicu-Tivadar, V, "Support-Vector Machine-Based Classifier of Cross-Correlated Phoneme Segments for Speech Sound Disorder Screening", 32th Medical Informatics Europe (MIE 2022) Conference, Vol. 294, pp. 455-459, Mai, 2022.

18. Varga, G; Stoicu-Tivadar, L; **Nicola, S**, "Serious Gaming and Artificial Intelligence in Rehabilitation of Rheumatoid Arthritis", 20th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2022), Vol. 295, pp. 562-565, Athens, Greece.

19. **Nicola, S**; Chirila, OS; Stoicu-Tivadar, L, "Comparison of Data Classification Results for Leap Motion Recovery Gestures", 20th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2022), Vol. 295, pp. 189-192, Athens, Greece.

7.3. Direcții de continuare a cercetării

Contribuțiile aduse în special în capitolul 4 și 5 al acestei teze de doctorat, permit extinderea cercetării. Astfel în viitor îmi propun să realizez următoarele:

- Construirea de noi gesturi LM utilizând doar seturile de date existente.
- Extinderea setului de date ce descriu gesturile LM astfel încât să se poată definii și alte gesturi și compararea rezultatelor obținute.
- Combinarea a două seturi de date diferite și aplicarea celor 12 modele de rețele utilizate inițial. Analizarea rezultatelor obținute.
- Creșterea calității gesturilor LM prin folosirea de rețele neuronale de tipul end–to- end, astfel încât în momentul când nu mai pot fii recunoscute gesturile făcute, rețeaua să preia atribuțiile dispozitivului LM și să continue gestul.
- Utilizarea noilor tehnologii MR și XR pentru a crea noi experiențe utilizatorilor ce folosesc gesturi în aplicații 3D.
- Aplicarea gesturilor construite pe noua generație a dispozitivului LM, Leap Motion Controller 2. Verificarea rezultatelor obținute prin acest dispozitiv.
- După cum se observă în lista de publicații, deja exprim un interes ridicat în procesarea semnalelor audio, a cuvintelor. Astfel în viitor doresc să aplic și algoritmii dezvoltați pentru procesarea semnalelor audio (algoritmi de cros corelare și auto corelare de semnale audio) în aplicații care se bazează pe interacțiuni multimodale în 3D.

Bibliografie

[MW23] Merriam Webster, https://www.merriam-webster.com/dictionary/gesture, ultima accesare: 10.06.2023

[DEX23] Dex online, https://dexonline.ro/definitie/gest, ultima accesare: 10.06.2023

[LM16] Leap Motion, https://www.ultraleap.com/, ultima accesare: 10.06.2023

[MAYO23] Mayo Armband, https://uncrate.com/myo-gesture-control-armband/, ultima accesare: 10.06.2023

[POL18] M. Polsinelli et al., "Hand movement parameters calculated by the LEAP based Virtual Glove," 2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA), Rome, Italy, 2018, pp. 1-6, doi: 10.1109/MeMeA.2018.8438764.

[AME16] S. Ameur, A. B. Khalifa and M. S. Bouhlel, "A comprehensive leap motion database for hand gesture recognition," 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Hammamet, Tunisia, 2016, pp. 514-519, doi: 10.1109/SETIT.2016.7939924.

[LI22] X. Li, R. Wang, B. Zhang and K. Wang, "Design of VR Experimental System Based on Leap Motion Gesture Recognition," 2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI), Zhuhai, China, 2022, pp. 305-309, doi: 10.1109/IWECAI55315.2022.00065.

[SHU19] Y Shu and Q Shu, "Application Research of VR Motion Capture System in MAYA Animation Production", Art Technology, vol. 32, no. 01, pp. 94, 2019.

[FIL18] I. A. S. Filho, "Gesture Recognition Using Leap Motion: A Machine Learning-based Controller Interface," SBC -Proceedings SBGames 2018, no. June, p. 4, 2018.

[SAA22] M. Saad, T. Yang and H. Zhou, "A Comparison of Bidirectional GRU and LSTM for Hand Gesture Recognition Using Leap Motion," 2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC), Beijing, China, 2022, pp. 1427-1433, doi: 10.1109/YAC57282.2022.10023591.

[JEN21] J. Jenkins and S. Rashad, "An Innovative Method for Automatic American Sign Language Interpretation using Machine Learning and Leap Motion Controller," 2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 2021, pp. 0633-0638, doi: 10.1109/UEMCON53757.2021.9666640.

[NAG16] D. Naglot and M. Kulkarni, "Real time sign language recognition using the leap motion controller," 2016 International Conference on Inventive

Computation Technologies (ICICT), Coimbatore, India, 2016, pp. 1-5, doi: 10.1109/INVENTIVE.2016.7830097.

[1AME20] S. Ameur, A. Ben Khalifa and M. S. Bouhlel, "Hand-Gesture-Based Touchless Exploration of Medical Images with Leap Motion Controller," 2020 17th International Multi-Conference on Systems, Signals & Devices (SSD), Monastir, Tunisia, 2020, pp. 6-11, doi: 10.1109/SSD49366.2020.9364244.

[CHU14] C. -H. Chuan, E. Regina and C. Guardino, "American Sign Language Recognition Using Leap Motion Sensor," 2014 13th International Conference on Machine Learning and Applications, Detroit, MI, USA, 2014, pp. 541-544, doi: 10.1109/ICMLA.2014.110.

[AVO19] D. Avola, M. Bernardi, L. Cinque, G. L. Foresti and C. Massaroni, "Exploiting Recurrent Neural Networks and Leap Motion Controller for the Recognition of Sign Language and Semaphoric Hand Gestures," in IEEE Transactions on Multimedia, vol. 21, no. 1, pp. 234-245, Jan. 2019, doi: 10.1109/TMM.2018.2856094.

[SME17] Q. De Smedt et al., "Shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset," Eurographics Workshop 3D Object Retrieval, 2017, pp. 1–7.

[WAN17] Q. Wang, Y. Wang, F. Liu and W. Zeng, "Hand gesture recognition of Arabic numbers using leap motion via deterministic learning," 2017 36th Chinese Control Conference (CCC), Dalian, China, 2017, pp. 10823-10828, doi: 10.23919/ChiCC.2017.8029083.

[WAN18] X. Wang, Z. Chen, X. Wang, Q. Zhao and B. Liang, "A Comprehensive Evaluation of Moving Static Gesture Recognition with Convolutional Networks," 2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Singapore, 2018, pp. 7-11, doi: 10.1109/ACIRS.2018.8467228.

[LUP16] R. G. Lupu, N. Botezatu, F. Ungureanu, D. Ignat and A. Moldoveanu, "Virtual reality based stroke recovery for upper limbs using leap motion," 2016 20th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 2016, pp. 295-299, doi: 10.1109/ICSTCC.2016.7790681.

[WEI17] X. Wei, Y. Huang, Z. Wu and C. Tang, "A method of rehabilitation training for finger pair movement based on multi — Leap motions," 2017 2nd International Conference on Information Technology (INCIT), Nakhonpathom, Thailand, 2017, pp. 1-6, doi: 10.1109/INCIT.2017.8257886.

[LI17] W. -J. Li, C. -Y. Hsieh, L. -F. Lin and W. -C. Chu, "Hand gesture recognition for post-stroke rehabilitation using leap motion," 2017 International Conference on Applied System Innovation (ICASI), Sapporo, Japan, 2017, pp. 386-388, doi: 10.1109/ICASI.2017.7988433.Utilizarea modelelor SVM si KNN

[LI18] X. Li, K. Wan, R. Wen and Y. Hu, "Development of Finger Motion Reconstruction System Based on Leap Motion Controller," 2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), Ottawa, ON, Canada, 2018, pp. 1-5, doi: 10.1109/CIVEMSA.2018.8439953.

[WAL19] H. Walugembe, C. Phillips, J. Requena-Carrión and T. Timotijevic, "Gesture Recognition in Leap Motion Using LDA and SVM," 2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE), London, UK, 2019, pp. 56-61, doi: 10.1109/iCCECE46942.2019.8941805.

[CAR20] L. Caro, E. Villablanca and B. Peralta, "A Virtual Reality Application using Hand Motion Tracking with Leap Motion Sensor," 2020 39th International Conference of the Chilean Computer Science Society (SCCC), Coquimbo, Chile, 2020, pp. 1-6, doi: 10.1109/SCCC51225.2020.9281210.

[SA-N19] P. Sa-nguannarm, T. Charoenpong, C. Chianrabutra and K. Kiatsoontorn, "A Method of 3D Hand Movement Recognition by a Leap Motion Sensor for Controlling Medical Image in an Operating Room," 2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP), Bangkok, Thailand, 2019, pp. 17-20, doi: 10.1109/ICA-SYMP.2019.8645985.

[KAV20] Kavian M, Nadian-Ghomsheh A. "Monitoring Wrist and Fingers Range of Motion using Leap Motion Camera for Physical Rehabilitation", International Conference on Machine Vision and Image Processing (MVIP), 2020, p. 1-6.

[RAK20] N. F. Rakib, N. H. Mahmood, N. Ramli, N. A. Zakaria and M. A. A. Razak, "Preliminary Results of Hand Rehabilitation for Post Stroke Patient using Leap Motion-based Virtual Reality," 2020 IEEE Student Conference on Research and Development (SCOReD), Batu Pahat, Malaysia, 2020, pp. 259-262, doi: 10.1109/SCOReD50371.2020.9250985.

[CHO18] Y. Cho, A. Lee, J. Park, B. Ko, N. Kim, "Enhancement of gesture recognition for contactless interface using a personalized classifier in the operating room", Computer Methods and Programs in Biomedicine, Volume 161, 2018, Pages 39-44, ISSN 0169-2607, https://doi.org/10.1016/j.cmpb.2018.04.003.

[XUE18] Y. Xue, L. Zhao, M. Xue and J. Fu, "Gesture Interaction and Augmented Reality based Hand Rehabilitation Supplementary System," 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2018, pp. 2572-2576, doi: 10.1109/IAEAC.2018.8577476.

[YAN20] Q. Yang, W. Ding, X. Zhou, D. Zhao and S. Yan, "Leap Motion Hand Gesture Recognition Based on Deep Neural Network," 2020 Chinese Control And Decision Conference (CCDC), Hefei, China, 2020, pp. 2089-2093, doi: 10.1109/CCDC49329.2020.9164723.

[BRO20] H. Brock, J. Ponce Chulani, L. Merino, D. Szapiro and R. Gomez, "Developing a Lightweight Rock-Paper-Scissors Framework for Human-Robot Collaborative Gaming," in IEEE Access, vol. 8, pp. 202958-202968, 2020, doi: 10.1109/ACCESS.2020.3033550.

[BHI22] N. M. Bhiri, S. Ameur, I. Jegham, M. A. Mahjoub and A. Ben Khalifa, "Fisher-HHT: A Feature Extraction Approach For Hand Gesture Recognition With a Leap Motion Controller," 2022 6th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), Sfax, Tunisia, 2022, pp. 1-6, doi: 10.1109/ATSIP55956.2022.9805899.

[2AME20] S. Ameur, A. Ben Khalifa, and M. S. Bouhlel, "A novel hybrid bidirectional unidirectional lstm network for dynamic hand gesture recognition with leap motion," Entertainment Computing, vol. 35, p. 100373, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1875952120300811

[DZI18] A. Dzikri and D. E. Kurniawan, "Hand Gesture Recognition for Game 3D Object Using The Leap Motion Controller with Backpropagation Method," 2018 International Conference on Applied Engineering (ICAE), Batam, Indonesia, 2018, pp. 1-5, doi: 10.1109/INCAE.2018.8579400.

[SUM19] S. Sumpeno, I. G. A. Dharmayasa, S. M. S. Nugroho and D. Purwitasari, "Immersive Hand Gesture for Virtual Museum using Leap Motion Sensor Based on K-Nearest Neighbor," 2019 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM), Surabaya, Indonesia, 2019, pp. 1-6, doi: 10.1109/CENIM48368.2019.8973273.

[KRI19] K. Kritsis, M. Kaliakatsos-Papakostas, V. Katsouros and A. Pikrakis, "Deep Convolutional and LSTM Neural Network Architectures on Leap Motion Hand Tracking Data Sequences," 2019 27th European Signal Processing Conference (EUSIPCO), A Coruna, Spain, 2019, pp. 1-5, doi: 10.23919/EUSIPCO.2019.8902973.

[BAS20] G. Bastas, K. Kritsis and V. Katsouros, "Air-Writing Recognition using Deep Convolutional and Recurrent Neural Network Architectures," 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), Dortmund, Germany, 2020, pp. 7-12, doi: 10.1109/ICFHR2020.2020.00013.

[SHI20] B.R. Shin, H.S. Son, S.P. Lee, H.S. Han, A Gesture Recognition System Using a Flexible Epidermal Tactile Sensor Based on Artificial Neural Network, International Conference on Robotics and Automation Sciences, Hong Kong, 2017.

[1MOS20] A. Moshkova, A. Samorodov, E. Ivanova and E. Fedotova, "High Accuracy Discrimination of Parkinson's Disease from Healthy Controls by Hand Movements Analysis Using LeapMotion Sensor and 1D Convolutional Neural Network," 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), Yekaterinburg,

Russia, 2020, pp. 0062-0065, doi: 10.1109/USBEREIT48449.2020.9117736.

[2MOS20] A. Moshkova, A. Samorodov, N. Voinova, A. Volkov, E. Ivanova and E. Fedotova, "Parkinson's Disease Detection by Using Machine Learning Algorithms and Hand Movement Signal from LeapMotion Sensor," 2020 26th Conference of Open Innovations Association (FRUCT), Yaroslavl, Russia, 2020, pp. 321-327, doi: 10.23919/FRUCT48808.2020.9087433.

[LI19] C. Li, A. Fahmy and J. Sienz, "Development of a Neural Network-Based Control System for the DLR-HIT II Robot Hand Using Leap Motion," in IEEE Access, vol. 7, pp. 136914-136923, 2019, doi: 10.1109/ACCESS.2019.2942648.

[LI20] H. Li, L. Wu, H. Wang, C. Han, W. Quan and J. Zhao, "Hand Gesture Recognition Enhancement Based on Spatial Fuzzy Matching in Leap Motion," in IEEE Transactions on Industrial Informatics, vol. 16, no. 3, pp. 1885-1894, March 2020, doi: 10.1109/TII.2019.2931140.

[CHE19] J. Chen, C. Liu, R. Cui and C. Yang, "Hand Tracking Accuracy Enhancement by Data Fusion Using Leap Motion and Myo Armband," 2019 IEEE International Conference on Unmanned Systems and Artificial Intelligence (ICUSAI), Xi'an, China, 2019, pp. 256-261, doi: 10.1109/ICUSAI47366.2019.9124812.

[HUO21] J. Huo, K. L. Keung, C. K. M. Lee and H. Y. Ng, "Hand Gesture Recognition with Augmented Reality and Leap Motion Controller," 2021 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, Singapore, 2021, pp. 1015-1019, doi: 10.1109/IEEM50564.2021.9672611.

[SME23] K. Sneha and R. Kayalvizhi, "Mid-Air Gesture Based Multi-Finger Control System For Paralyzed Patients Using Leap Motion," 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2023, pp. 1-6, doi: 10.1109/ICCCI56745.2023.10128585.

[KUM17] P. Kumar, R. Saini, P. P. Roy and D. P. Dogra, "Study of Text Segmentation and Recognition Using Leap Motion Sensor," in IEEE Sensors Journal, vol. 17, no. 5, pp. 1293-1301, 1 March1, 2017, doi: 10.1109/JSEN.2016.2643165.

[ERD16] K. Erdoğan, A. Durdu and N. Yilmaz, "Intention recognition using leap motion controller and Artificial Neural Networks," 2016 International Conference on Control, Decision and Information Technologies (CoDIT), Saint Julian's, Malta, 2016, pp. 689-693, doi: 10.1109/CoDIT.2016.7593646.

[RAD21] U. Radhakrishnan, F. Chinello and K. Koumaditis, "Immersive Virtual Reality Training : Three Cases from the Danish Industry," 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Lisbon, Portugal, 2021, pp. 1-5, doi: 10.1109/VRW52623.2021.00008.

[CHE17] L. Chen, T. W. Dayz, W. Tangx, N. W. John, Recent Developments and Future Challenges in Medical Mixed Reality, 16th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 3 Aug, 2017.

[BIR18] J. Birt, Z. Stromberga, M. Cowling, C. Moro, Mobile Mixed Reality for Experiential Learning and Simulation in Medical and Health Sciences Education, Serious Games and Applications for Health (2018).

[PIN20] C. Pinter et al., "SlicerVR for Medical Intervention Training and Planning in Immersive Virtual Reality," in IEEE Transactions on Medical Robotics and Bionics, vol. 2, no. 2, pp. 108-117, May 2020, doi: 10.1109/TMRB.2020.2983199.

[CHH19] V. Chheang et al., "Collaborative Virtual Reality for Laparoscopic Liver Surgery Training," 2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), San Diego, CA, USA, 2019, pp. 1-17, doi: 10.1109/AIVR46125.2019.00011.

[HTCV23] HTC Vive, https://www.vive.com/us/, ultima accesare: 12.06.2023

[EVE22] T. Everson, M. Joordens, H. Forbes and B. Horan, "Virtual Reality and Haptic Cardiopulmonary Resuscitation Training Approaches: A Review," in IEEE Systems Journal, vol. 16, no. 1, pp. 1391-1399, March 2022, doi: 10.1109/JSYST.2020.3048140.

[GRU21] Gruenewald et al., "Virtual Reality Training Application to Prepare Medical Student's for Their First Operating Room Experience," 2021 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), Taichung, Taiwan, 2021, pp. 201-204, doi: 10.1109/AIVR52153.2021.00045.

[LAV19] Lavoué E, Monterrat B, Desmarais M and George S, Adaptive Gamification for Learning Environments, IEEE Transactions on Learning Technologies. Jan.-March 2019;12(1):16-28

[RAJ19] P. Rajeswaran, J. Varghese, P. Kumar, J. Vozenilek and T. Kesavadas, "AirwayVR: Virtual Reality Trainer for Endotracheal Intubation," 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 2019, pp. 1345-1346, doi: 10.1109/VR.2019.8797998.

[CHA22]Chang, A.-H.; Lin, P.-C.; Lin, P.-C.; Lin, Y.-C.; Kabasawa, Y.; Lin, C.-Y.; Huang, H.-L. Effectiveness of Virtual Reality-Based Training on Oral Healthcare for Disabled Elderly Persons: A Randomized Controlled Trial. J. Pers. Med. 2022, 12, 218. https://doi.org/10.3390/jpm12020218

[WU19] X. Wu, H. Liu, J. Zhang and W. Chen, "Virtual reality training system for upper limb rehabilitation," 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA), Xi'an, China, 2019, pp. 1969-1974, doi: 10.1109/ICIEA.2019.8834288.

[GEO21] C. Georgiadis et al., "A remote rehabilitation training system using Virtual Reality," 2021 6th South-East Europe Design Automation, Computer

Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), Preveza, Greece, 2021, pp. 1-4, doi: 10.1109/SEEDA-CECNSM53056.2021.9566227.

[NAG21] Y. Nagashima, D. Ito, R. Ogura, T. Tominaga and Y. Ono, "Development of Virtual Reality-based Gait Training System Simulating Personal Home Environment," 2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Mexico, 2021, pp. 5764-5767, doi: 10.1109/EMBC46164.2021.9631077.

[LIU20] Y. Liu, Q. Sun, Y. Tang, Y. Li, W. Jiang and J. Wu, "Virtual reality system for industrial training," 2020 International Conference on Virtual Reality and Visualization (ICVRV), Recife, Brazil, 2020, pp. 338-339, doi: 10.1109/ICVRV51359.2020.00091.

[JIA21] Z. Jiang, Y. Yang, Q. Yuan, P. Leng, Y. Liu and Z. Pan, "Virtual Reality Training Environment for Electric Systems," 2021 IEEE 7th International Conference on Virtual Reality (ICVR), Foshan, China, 2021, pp. 314-318, doi: 10.1109/ICVR51878.2021.9483825.

[UNIT23] Unity, https://unity.com/, ultima accesare 12.06.2023.

[VUFO23] Vuforia, https://developer.vuforia.com/, ultima accesare 12.06.2023.

[TON20] Tony Liao, Pamara F. Chang, SongYi Lee, Chapter 6 - Augmented reality in health and medicine: A review of augmented reality application for health professionals, procedures, and behavioral interventions, Editor(s): Jihyun Kim, Hayeon Song, Technology and Health, Academic Press, 2020,Pages 109-128, ISBN 9780128169582, https://doi.org/10.1016/B978-0-12-816958-2.00006-X.

[MOA23] Moayad Aloqaily, Ouns Bouachir, Fakhri Karray, Chapter 3 - Digital twin for healthcare immersive services: fundamentals, architectures, and open issues, Editor(s): Abdulmotaleb El Saddik, Digital Twin for Healthcare, Academic Press, 2023, Pages 39-71, ISBN 9780323991636, https://doi.org/10.1016/B978-0-32-399163-6.00008-1.

[SCH22] J. Schild, G. Carbonell, A. Tarrach and M. Ebeling, "ViTAWiN - Interprofessional Medical Mixed Reality Training for Paramedics and Emergency Nurses," 2022 IEEE 10th International Conference on Serious Games and Applications for Health(SeGAH), Sydney, Australia, 2022, pp. 1-8, doi: 10.1109/SEGAH54908.2022.9978566.

[POP20] D. -L. Popa, M. -L. Mocanu, R. -T. Popa, L. -F. Bărbulescu, L. N. Bărbulescu and C. Ciobîrcă, "Augmented and mixed reality for collaborative cardiac planning using 3D and 4D medical imaging data," 2020 24th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 2020, pp. 384-389, doi: 10.1109/ICSTCC50638.2020.9259677.

[HEN22] B. Hensen and R. Klamma, "A Mixed Reality Teaching Course for Formal Higher Education," 2022 8th International Conference of the Immersive

Learning Research Network (iLRN), Vienna, Austria, 2022, pp. 1-5, doi: 10.23919/iLRN55037.2022.9815961.

[WAN19] P. Wang et al., "An MR Remote Collaborative Platform Based on 3D CAD Models for Training in Industry," 2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Beijing, China, 2019, pp. 91-92, doi: 10.1109/ISMAR-Adjunct.2019.00038.

[MOR18] Moreta-Martinez R, García-Mato D, García-Sevilla M, Pérez-Mañanes R, Calvo-Haro J, Pascau J. Augmented reality in computer-assisted interventions based on patient-specific 3D printed reference. Healthc Technol Lett. 2018 Sep 14;5(5):162-166. doi: 10.1049/htl.2018.5072. PMID: 30464847; PMCID: PMC6222179.

[THO17] Thomas W. Edgar, David O. Manz, Chapter 4 - Exploratory Study, Editor(s): Thomas W. Edgar, David O. Manz, Research Methods for Cyber Security, Syngress, 2017, Pages 95-130, ISBN 9780128053492, https://doi.org/10.1016/B978-0-12-805349-2.00004-2

[CHA20] Debjani Chakraborty, Ahona Ghosh, Sriparna Saha, Chapter 2 - A survey on Internet-of-Thing applications using electroencephalogram, Editor(s): Valentina Emilia Balas, Vijender Kumar Solanki, Raghvendra Kumar, Emergence of Pharmaceutical Industry Growth with Industrial IoT Approach, Academic Press, 2020, Pages 21-47, ISBN 9780128195932, https://doi.org/10.1016/B978-0-12-819593-2.00002-9.

[WAN23] Alex X. Wang, Stefanka S. Chukova, Binh P. Nguyen, Ensemble k-nearest neighbors based on centroid displacement, Information Sciences, Volume 629, 2023, Pages 313-323, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2023.02.004.

[OOI17] Melanie Po-Leen Ooi, Hong Kuan Sok, Ye Chow Kuang, Serge Demidenko, Chapter 19 - Alternating Decision Trees, Editor(s): Pijush Samui, Sanjiban Sekhar, Valentina E. Balas, Handbook of Neural Computation, Academic Press, 2017,Pages 345-371, ISBN 9780128113189, https://doi.org/10.1016/B978-0-12-811318-9.00019-3.

[VIJ23] V. Vijay and P. Verma, "Variants of Naïve Bayes Algorithm for Hate Speech Detection in Text Documents," 2023 International Conference on Artificial Intelligence and Smart Communication (AISC), Greater Noida, India, 2023, pp. 18-21, doi: 10.1109/AISC56616.2023.10085511.

[NAS17] A. T. Nasser and N. Dogru, "Signature recognition by using SIFT and SURF with SVM basic on RBF for voting online," 2017 International Conference on Engineering and Technology (ICET), Antalya, Turkey, 2017, pp. 1-5, doi: 10.1109/ICEngTechnol.2017.8308208.

[KHA19] G. Khanvilkar and D. Vora, "Smart Recommendation System Based on Product Reviews Using Random Forest," 2019 International Conference on Nascent Technologies in Engineering (ICNTE), Navi Mumbai, India, 2019, pp. 1-9, doi: 10.1109/ICNTE44896.2019.8945855.

[ALM20] M. Almaghrabi and G. Chetty, "Improving Sentiment Analysis in Arabic and English Languages by Using Multi-Layer Perceptron Model (MLP)," 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), Sydney, NSW, Australia, 2020, pp. 745-746, doi: 10.1109/DSAA49011.2020.00095.

[CHI19] Jen-Tzung Chien, Chapter 7 - Deep Neural Network, Editor(s): Jen-Tzung Chien, Source Separation and Machine Learning, Academic Press, 2019, Pages 259-320, ISBN 9780128177969, https://doi.org/10.1016/B978-0-12-804566-4.00019-X.

[NIC16] Nicola, S; Stoicu-Tivadar, L; Virag, I; Crisan-Vida, M, "Leap motion supporting medical education", in Proc. 12th IEEE International Symposium on Electronics and Telecommunications (ISETC 2016), pp. 153-156, Timișoara, România, WOS: 000390717800035 (ISI)

[1NIC20] Nicola, S; Chirila, OS; Stoicu-Tivadar, L, "Gesture Classification for a Hand Controller Device Using Neural Networks", 30th Medical Informatics Europe (MIE 2020) Conference, Vol. 270, pp. 756-760, APR, 2020, WOS:000630065600058 (ISI)

[2NIC20] Nicola, S; Stoicu-Tivadar, L "Evaluating Interactivity in VR Healthcare Applications Using Analytics", in Proc. 18th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2020), Vol. 272, pp. 225-228, Athens, Greece, WOS:000560388600073 (ISI)

[1NIC18] Nicola, S; Stoicu-Tivadar, L, "Mixed Reality Supporting Modern Medical Education", in Proc. Special Topic Conference of the European-Fedeeration-for-Medical-Informatics (EFMI STC 2018), Vol. 255, pp. 242-246, Zagreb, Croatia, WOS: 000455957400047 (ISI)

[2NIC18] Nicola, S; Stoicu-Tivadar, L; Patrascoiu, A, "VR for Education in Information and Tehnology: application for Bubble Sort", in Proc. 13th International Symposium on Electronics and Telecommunications (ISETC 2018), pp. 343-346, Timișoara, România, WOS: 000463031500076 (ISI)

[1NIC17] Nicola, S; Handrea, FL; Crisan-Vida, M; Stoicu-Tivadar, L, "DNA Encoding Training Using 3D Gesture Interaction", in Proc. Special Topic Conference of the European-Fedeeration-for-Medical-Informatics (EFMI STC 2017), Vol. 244, pp. 63-67, Tel Aviv, Israel, WOS: 000450270500013 (ISI)

[2NIC17] Nicola, S; Virag, I; Stoicu-Tivadar, L, "VR Medical Gamification for Training and Education", in Proc. 11th Annaul Conference on Health Informatics Meets eHealth (eHealth 2017), Vol. 236, pp. 97- 103, Schloss Schonbrunn, Austria, WOS: 000426828000013 (ISI)

[3NIC18] Nicola, S; Lupșe, OS, Stoicu-Tivadar, L, "Novel Gesture Interaction Using Leap Motion in 3D Applications", in Proc. 12th International Symposium on Applied Computational Intelligence and Informatics (SACI 2018), Timișoara, Romania, May 2018, pp. 113-118, WOS:000448144200020 (ISI)

[NIC18] Nicola, S; Stoicu-Tivadar, L, "Hand Rehabilitation Using a 3D Environment and Leap Motion Device", Studies in health technology and informatics (ICIMTH 2018), Vol. 251, pp. 43-46, ISSN: 0926-9630

[NIC19] Nicola, S; Chirila, OS.; Stoicu-Tivadar, L, "Enhancing Precision in Gesture Detection for Hand Recovery fter Injury Using Leap Motion and Machine Learning", in Proc. 18th International Conference on Informatics, Management and Technology in Healthcare (ICIMTH 2019), Vol. 262, pp. 320-323, Athens, Greece, WOS:000560388600073 (ISI)