Departamentul Automatică și Informatică Industrială
Facultatea Automatică şi Calculatoare
Universitatea Nationala de Stiinta și Tehnologie
POLITEHNICA din Bucureşti
Splaiul Independenţei 313, 060042, Bucureşti, România
Sala ED 412, Tel. 021/402.92.69
www.aii.pub.ro, email: secretariat.aii@upb.ro

Sesiunea de Comunicări Şiinţifice Studenţeşti

Ediţia 2024

Editare de imagine bazată pe hand recognition

Autor : Daniel-Andrei NELEPTCU, Facultatea de Automatică și Calculatoare, Anul III, Grupa 332AB

Adresa e-mail : neleptcu@gmail.com

Îndrumător ştiinţific : Conf. dr. ing. Ștefan MOCANU

Cuprins:

1. Introducere

Tehnologia se dezvoltă din ce în ce mai mult, iar o data cu aceasta dezvoltare, aplicațiile interactive devin mai populare. Din acestea fac parte și aplicațiile de editare a imaginilor ce au un efect vital în expresivitatea umană: de la editarea fotografiilor pentru a avea un impact mai mare pe rețelele de socializare, la ascunderea imperfecțiunilor, extinderea emoției generată de o fotografie prin aplicarea anumitor efecte potrivite contextului; Pe langa acestea, atât aplicațiile de design și desen digital și-au făcut simțită prezența prin animațiile și desenele dezvoltate ca și calitate, dar și cerinta și consumul acestora de către public, în special în perioada pandemiei COVID.

Fară să realizăm am interacționat cu editarea de fotografii. De la o simpla încărcare a fotografiilor pe un site de socializare sau utilizarea telefonului și a aparatului foto pentru a face fotografii, deja se produce o varietate de procedee ce țin de editarea fotografiilor atât la nivel calitativ cât și cantitativ. Printre efectele de bază cu care interactțonam se află manipularea luminozității, a rezoluției, a calității și miscării imaginii prin stabilizări atât fizice cât și digitale, schimbarea formatelor și spatiilor de culoare, compresia imaginii și multe altele.

Editarea fotografiilor reprezintă un domeniu care se află la limita dintre creativitate și tehnicalitate. De foarte multe ori, anumite rezultate ce pornesc dintr-o dorință de creativitate au în spate multe tehnicalități matematice, cu artificii ce tin de „păcălirea” ochiului uman și manipularea algoritmică a componentelor imaginii.

Aceste procedee sunt bine stabilite, iar subiectivitatea domeniului vine din variația rezultatului final pe care îl doreste utilizatorul. Din cauza subiectivității, introducerea învățării automate într-un domeniu artistic stârneste controverse, iar editarea imaginilor este o legătură ce ameliorează aceste tensiuni. Prin multitudinea de filtre de edge-detection, image-warping, color-isolation, învățarea automata se folosește de principiile editării de imagine pentru a obține rezultate cu utilitate reală: mașini autonome, imagistica medicală, detectarea obiectelor din imagini, rezultate ce influențeaza oamenii și în multe cazuri imbunătățesc calitatea vietii.

2. Prezentarea domeniului din care face parte lucrarea

2.1. Editarea imaginilor

Image (Oxford Dictionary) - representation of the external form of a person or thing în art

Digital (Collins Dictionary) - representation of information using a finite number of values, usually represented în a computer through binary values.

Imaginile digitale sunt reprezentarea digitală a unei imagini, compuse din elemente de baza numite pixeli, fiecare pixel având o valoare finită și discretă ce descrie o culoare percepută de om, acestia fiind grupati in cadre dreptunghiulare descrise de înălțime și lățime.

Editarea imaginilor este un domeniu în continuă dezvoltare, care se ocupa cu modificarea și îmbunătățirea aspectului, compoziției și calității imaginilor digitale. În aceasta practica sunt implicate diverse tehnici și instrumente software pentru a realiza diferite scopuri, de la corecția unor imperfecțiuni minore până la transformări complexe. Editarea imaginilor în ziua de astăzi este versiunea digitală a fostelor „camere întunecate" folosite pentru developarea și procesarea filmului. în acele camere, tehnicile de editare implicau folosirea chimicalelor, suprapunerea fizică a negativelor pentru a obține poze mai clare sau chiar alterarea fizică a fotografiei. În ziua de astăzi editarea imaginilor a ajuns la un nivel mult mai înalt, dezvoltându-se în totalitate digital, cu o varietate de tehnici și opțiuni. Principalele tehnici de editare implică selecția unei porțiuni din imagine, îndreptarea, modificarea ratiei dintre înălțime și lățime, modificarea cantității de lumină, reducerea sau creșterea educată a zgomotului, modificarea parametrilor ce influențează culoarea imaginii. De la aceste tehnici de bază se pot produce efecte mult mai complexe cum ar fi eliminarea fundalului unei imagini, reliefarea unui obiect, evidențierea anumitor obiecte descrise de caracteristici cum ar fi contur și intensitatea culorii, manipularea umbrelor și eliminarea obiectelor nedorite din imagine. Aplicând principii de învățare automată ce pornesc de la conceptele de baza ale editării de imagine, domeniul se diversifică mult mai mult ajungând la aplicații practice ce depasesc spectrul aprecierii subiective a unei fotografii/imagini.

2.2. Machine learning

Machine learning (prin traducere: invățare automată) este un subdomeniu în inteligența artificială care se ocupa cu dezvoltarea și studiul algoritmilor statistici care învata din seturi de date și generalizează procedeele propuse pentru seturi de date pe care nu le-au testat, astfel executând procese fără a avea instrucțiuni explicite. Machine learning-ul își găsește aplicabilitate în multe domenii, cum ar fi : procesarea limbajului natural, computer vision, recunoașterea vorbirii, filtrare de email-uri, robotică și interpretare de semnale, agricultură, afaceri și piața de acțiuni sau chiar medicină. Chiar dacă nu tot machine learning-ul este bazat pe statistică, statistica computațională este unul din cei mai importanți piloni ai acestuia.

2.3. Computer vision

Computer vision (prin traducere: vedere computerizată) reprezintă o serie de procese pentru a prelua, procesa, analiza și învăța din imagini digitale folosind machine learning. Analogul procesului de computer vision este mecanismul ochi - creier. Acesta preia imagini și pe baza experiențelor anterioare și a anilor de evoluție a speciei reușește să clasifice obiecte, distanțe, să asocieze forme cu informație și să țnvete pe baza experiențelor vizuale. Vederea computerizată folosește machine learning pentru a învata din o multitudine de seturi de date să asocieze trăsături obiective ale obiectelor cu obiectele în sine ajungând la abilități comparabile și chiar mai bune decât mecanismul ochi - creier.

Printre domeniile la baza cărora se află computer vision se număra: imagistica și analiza medicală, securitate și recunoaștere facială, mașini autonome, traducerea textului în timp real, generarea de imagini pe baza altor imagini, detecția emoțiilor, detecția limbajului semnelor, ajutorul persoanelor cu deficiențe de vedere.

Computer vision reprezintă un pas important în dezvoltarea tehnologiei cu care interacționăm, având un potențial imens de a transforma lumea înconjurătoare. Cu toate acestea, este esențial să abordăm cu atenție aspectele etice și de confidențialitate, dar si bias-ul pe care un algoritm de acest tip îl poate genera în urma antrenării pe un set de date subiectiv și restrâns.

3. Motivație

La începutul anului 2024 Neuralink, o companie americană de neuro-tehnologie care dezvoltă interfețe creier-calculator implantabile, anunță un avans mare în ceea ce privește relația dintre medicină și tehnologie : *Nolan Arbaugh* a fost primul pacient al Neuralink ce a primit un implant cu ajutorul căruia poate controla cursorul de pe ecran folosindu-se de propriile gânduri. Deși mai există cazuri asemănătoare referitoare la controlarea unui cursor cu ajutorul unui implant sau recunoașterea semnalelor trimise de creier pentru a genera text, evenimentul a fost mediatizat datorită popularității companiei Neuralink, reaprinzând astfel interesul publicului față de implicarea tehnologiei, respectiv a invățării automate în medicină și recuperarea pacienților cu dizabilități.

Majoritatea aplicațiilor de editare de imagine nu au codul sursa disponibil, cu limitări în ceea ce priveste posibilitățile și dorintele unui utilizator. Aplicația neputând fi alterată în acest caz, utilizatorul este forțat să se multumească cu ceea ce este disponibil în momentul respectiv în aplicație. Deși editarea de imagine clasică presupune îmbunătățirea calității imaginii prin aplicarea diverselor procedee, din aceeași categorie de procedee fac parte și efectele ce nu produc o satisfacție vizuală generica. Efecte precum ar fi edge-detection, warping, color-swapping nu sunt implementate în mod direct pentru a vizualiza cum functioneaza, ci sunt folosite în implementari mai complexe cum ar fi eliminarea fundalului, selectarea inteligenta a obiectelor, pozitionarea imaginilor în poligoane customizate.

Plecând de la aceste două motive am decis să dezvolt o aplicație de editare a imaginilor ce își propune sa depaseasca limitarile date de aspectul inchis al transparentei codului și a lipsei efectelor intermediare, cu posibilitatea de a controla cursorul prin intermediul recunoașterii mâinii în fața camerei, oferind astfel și persoanelor ce nu pot manipula într-un mod clasic un mouse să beneficieze de aceasta.

4. Prezentarea aplicației și descrierea functionalitatilor

Aplicația este împărțită în două module principale:

- Modul editare imagine și desen
- Modul recunoastere a mâinii și manipulare cursor

4.1. Modulul de editare imagine și desen

Modulul de editare imagine și desen este o aplicație cu interfață grafica, care la lansare intampina utilizatorul cu un ecran negru și doua meniuri „File” și „Extra” asemănător *Fig.1.1.*

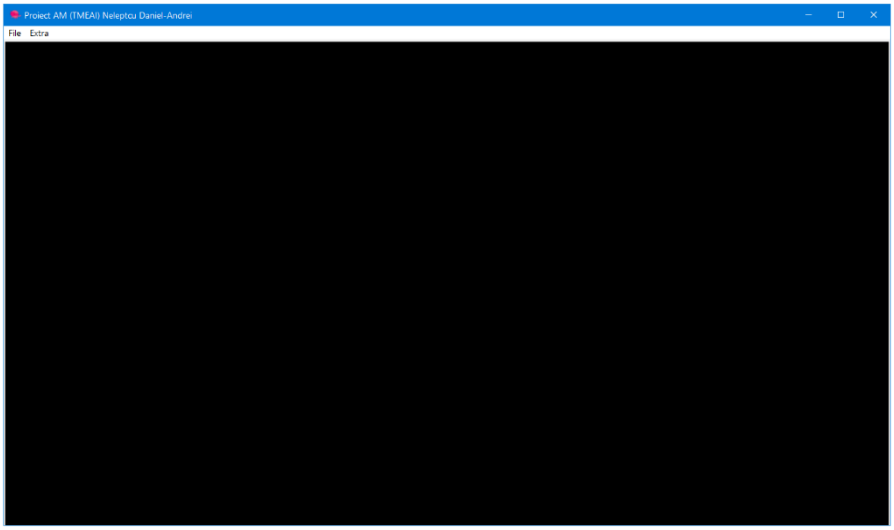

Fig.1.1. Ecranul principal al aplicației

Pentru a avansa în aplicație, utilizatorul are la dispoziție cele două meniuri menționate mai sus. La apăsarea meniului *„File”*, utilizatorul este întâmpinat cu meniul drop-down din *Fig.1.2.*

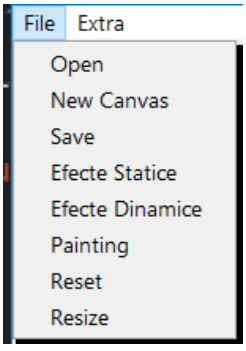

Fig.1.2. Meniul File

Open – se deschide o fereastră de dialog ce permite utilizatorului să aleagă un fișier de tip imagine pe care dorește să îl prelucreze. Directorul default este locatia *Desktop*.

New Canvas – se deschide o casetă de dialog ce permite utilizatorului selectarea unei rezoluții pentru a crea o planșă de desen cu fundal alb.

Save – se deschide o fereastră în care utilizatorul poate specifica calitatea imaginii în procente. La apăsarea butonului *„Proceed”* se deschide o fereastră de dialog ce permite utilizatorului să salveze modificările aplicate imaginii într-un fișier de tip imagine. Directorul default este locatia *Desktop*.

Efecte Statice – se deschide o nouă fereastra *[Fig.2.1.]* ce înglobează mai multe efecte statice cu intensitate prestabilită și unelte utilitare:

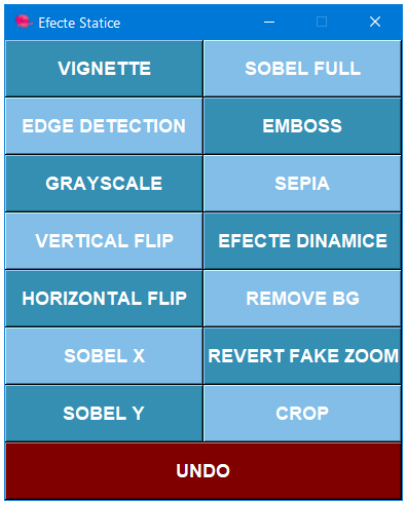

Fig.2.1. Efecte Statice

- *Vignette* – Centrează luminozitatea imaginii reducând din luminozitatea marginilor *[Fig.2.2.]*

Fig.2.2. Vignette Filter

- *Edge Detection* – Detectează marginile obiectelor folosind Canny edge detector *[Fig.2.3.]*

Fig.2.3. Edge Detection - Canny

- *Grayscale* – Imaginea din color în alb negru *[Fig.2.4.]*

Fig.2.4. Grayscale

- *Vertical Flip* – Oglindire față de axa Ox la jumătatea imaginii *[Fig.2.5.]*

Fig.2.5. Vertical Flip

- *Horizontal Flip* – Oglindire față de axa Oy la jumătatea imaginii *[Fig.2.6.]*

Fig.2.6. Horizontal Flip

- *Sobel X* – Aplicarea unui filtru de edge detection pe axa Ox *[Fig.2.7.]*

Fig.2.7. Sobel X

- *Sobel Y* – Aplicarea unui filtru de edge detection pe axa Oy *[Fig.2.8.]*

Fig.2.8. Sobel X

- *Sobel Full* – Aplicarea celor două filtre Sobel X și Sobel Y *[Fig.2.9.]*

Fig.2.9. Sobel Full

- *Emboss* – Efect de reliefare a unei imagini *[Fig.2.10.]*

Fig.2.10. Emboss

- *Sepia* – Transformarea unei fotografii în tonuri de rosu și maro *[Fig.2.11.]*

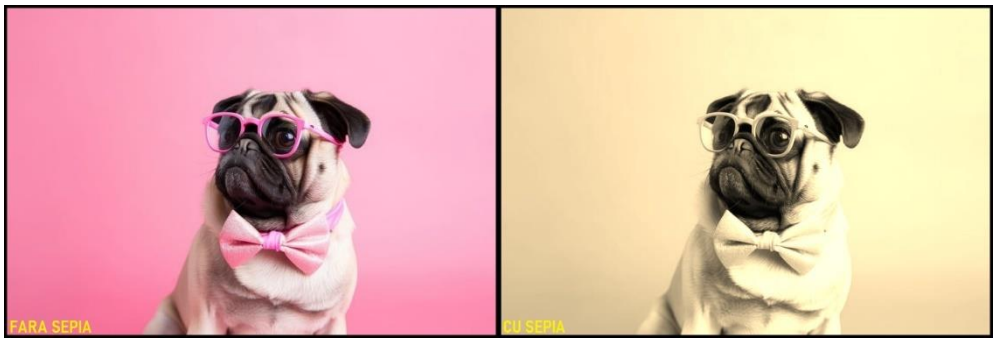

Fig.2.11. Sepia

- *Remove BG* – Eliminarea fundalului și păstrarea subiectelor din imagine *[Fig.2.12.]*

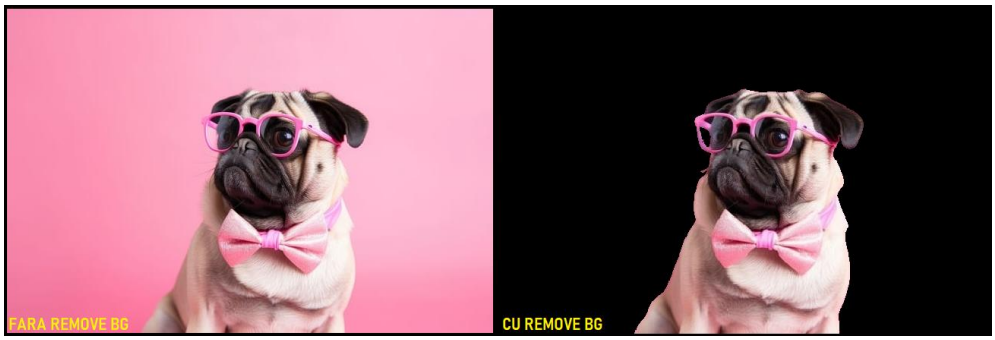

Fig.2.12. Remove BG

- *Crop* – Decuparea unei porțiuni dreptunghiulare din imagine *[Fig.2.13.]*

Fig.2.13. Crop

- *Revert Fake Zoom* – Utilizarea efectului de Crop pentru a aplica efecte pe zona decupată, reaplicând aceasta zonă peste imaginea anterioară *[Fig.2.14.]*

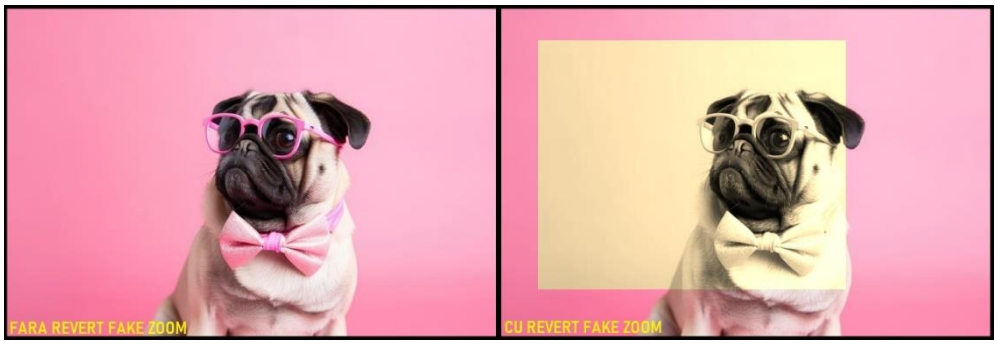

Fig.2.14. Revert Fake Zoom

- *Undo* – Anularea ultimei modificări facute (posibilitate de undo în cascadă)

Efecte Dinamice – se deschide o nouă fereastră *[Fig.3.1.]* ce inglobează mai multe efecte dinamice cu intensitate variabilă prin intermediul unor slidere. Efectele pot fi observate în *[Fig.3.2]* și *[Fig.3.3.]:*

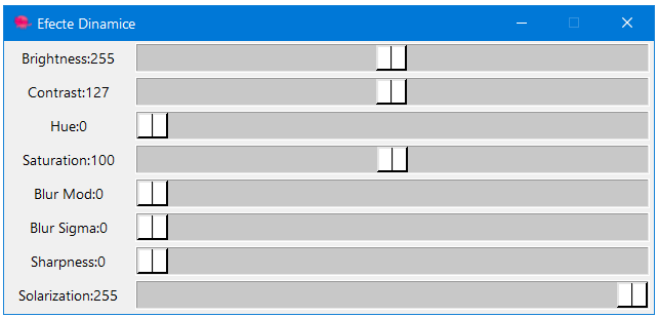

Fig.3.1. Efecte Dinamice

- *Brightness* – modifica nivelul de luminozitate al imaginii : factorului de luminozitate 0 ii corespunde imaginea neagra, factorului de luminozitate 512 ii corespunde imaginea alba.
- *Contrast* - modifică diferența de luminozitate / culoare între regiunile imaginii.
- *Hue* - izolează nuanța pură a unei culori de bază.
- *Saturation* – modifică intensitatea culorilor de bază.
- *Blur Mod* – factor de putere pentru filtrul care incețoșează imaginea.
- *Blur Sigma* – factor de putere pentru incețoșarea imaginii pe axa Ox.
- *Sharpness* – modifică claritatea / ascuțimea detaliilor dintr-o imagine.
- *Solarization* – inversarea tonurilor de culoare (specifică filmului expus la soare).

Fig.3.2. Efecte Dinamice

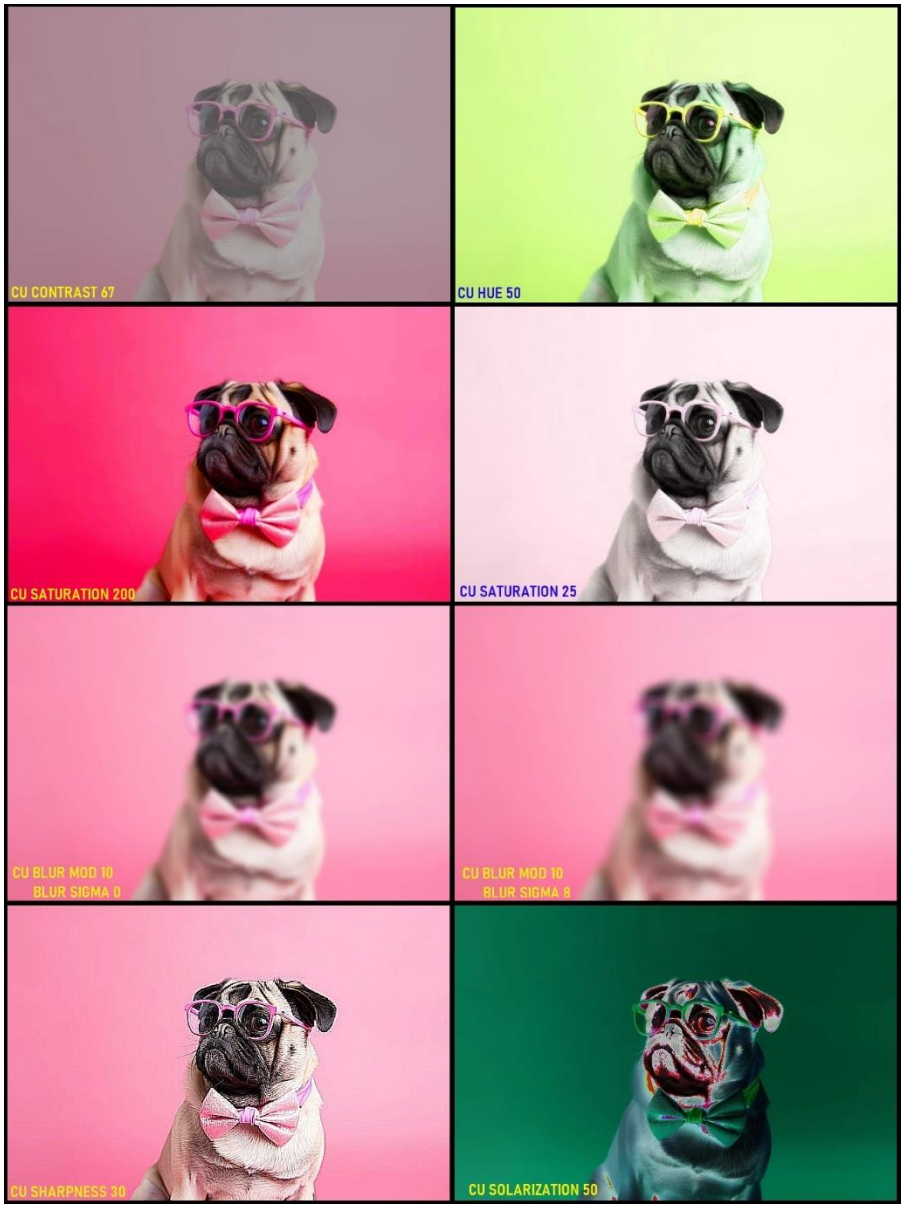

Fig.3.3. Efecte Dinamice

Efectele dinamice pot fi aplicate în combinație cu alte efecte dinamice și efecte statice, inclusiv cu efectul de Revert Fake Zoom.

Painting – se deschide o nouă fereastră *[Fig.4.1.]* ce prezinta o unealtă de tip Paint, cu care utilizatorul poate desena folosind cele 3 brush-uri predefinite: Default, Star, Spray. Pentru a activa modul de desenare utilizatorul va trebui să bifeze căsuța „Painting Brush”. Se pot selecta dimensiunea brush-ului, culoarea, pentru brush-ul Star se poate selecta densitatea și unghiul de formare al stelei, iar pentru brush-ul Spray se poate selecta densitatea spray-ului. Efectele acestor brush-uri pot fi observate în *[Fig.4.2.]*

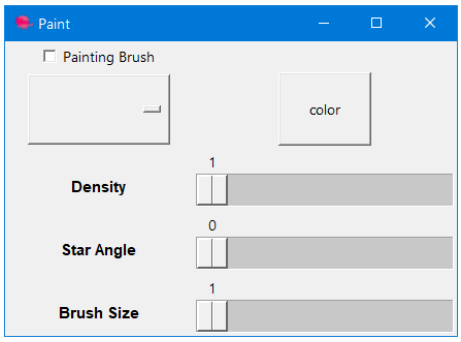

Fig.4.1. Paint tool

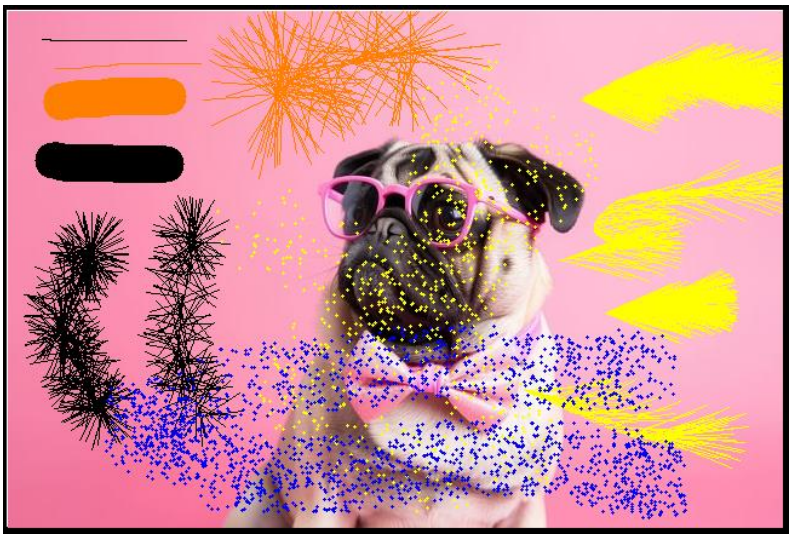

Fig.4.2. Brush-urile Default, Star, Spray cu diverse culori și

Reset – permite utilizatorului sa revină la imaginea inițial deschisa, ignorând toate modificările făcute până atunci.

Resize – se deschide o casetă de dialog în care utilizatorul poate selecta un factor de downscaling sau upscaling al imaginii, afișându-se noua rezoluție a acesteia. Acest efect interacționeaza cu toate efectele menționate anterior.

Meniul Custom – Prezintă utilizatorului opțiuni de a introduce propriile modificări asupra imaginii, introducând noi nuclee de convoluție și funcții.

4.2. Modulul de recunoastere a mâinii și manipulare a cursorului

Utilizatorul poate selecta din meniul Extra următoarele opțiuni:

- *Help* – Scurtă descriere a întregii aplicații.
- *Start Mouse* - Pornirea modulului de recunoaștere a mâinii și manipulare a cursorului.
- *Stop Mouse* – Oprirea modulului de recunoaștere a mâinii și manipulare a cursorului.

Pentru confirmarea intenției dupa lansarea opțiunii *„Start Mouse”* utilizatorul trebuie să țină palma deschisă în fața camerei pană la lansarea modulului. După lansarea acestuia, va fi întâmpinat cu o fereastră ce acceseaza camera web și o reprezentare grafică a interpretarii imaginii, respectiv a mâinii, de catre program. Pentru mișcarea cursorului, utilizatorul se va folosi de degetul arătător, ținând închise celelalte degete. Pentru a simula un click, utilizatorul va ridica o scurta perioadă de timp degetul mijlociu ținand degetul arătător ridicat. Pentru a simula mișcarea cursorului în timp ce click stânga este apasat, utilizatorul va ține doar degetul arătător și degetul mic ridicate. Modulul se va opri la selectarea opțiunii *„Stop Mouse”* din meniul *„Extra”* sau la închiderea modulului de editare imagine și desen. Interpretarea imaginii poate fi văzută în *[Fig.5.1.]*.

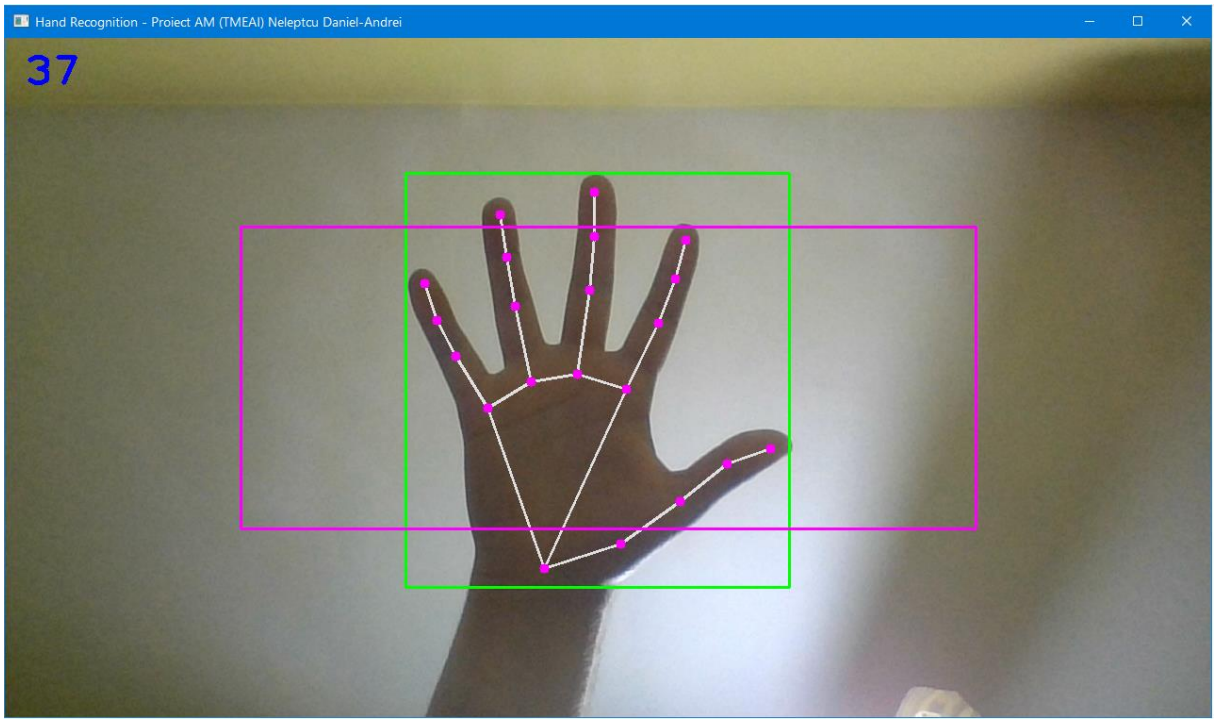

Fig.4.2. Modul de recunoastere a mâinii și manipulare a cursorului

5. Prezentarea implementarii

Lucrarea este realizată în limbajul Python 3.7, structurată în 3 fișiere ce descriu cele doua module implementate:

- Modulul de editare imagine și desen conține fișierele EditingApp.py
- Modulul de recunoaștere a mâinii și manipulare a cursorului conține fișierele VirtualMouse.py și HandDetectionModule.py

Datorită includerii a 4 pachete de procesare a imaginilor: *skimage, PIL, OpenCV, tkinter.Image*, respectiv a pachetelor *mediapipe* și *autopy* pentru controlul cursorului, respectarea cerințelor de performanță și dependințele între acestea au fost satisfăcute folosind *Python 3.7*.

5.1. Modulul de editare imagine și desen

Aplicația principală beneficiază de o interfață grafică implementată folosind pachetul tkinter. La baza programului se află fereastra root ce prezintă două meniuri ce conțin submeniuri cu comenzi asociate și un canvas cu fundalul negru. Pentru a păstra o standardizare a prezentării imaginilor generice, formatul de bază al canvas-ului este 16:9 cu o rezoluție de 1400x788 pixeli pentru imaginea de preview.

Meniul File conține submeniurile: *Open, New Canvas, Save, Efecte Statice, Efecte Dinamice, Painting, Reset, Resize*. Fiecare meniu are asociată o funcție specifică.

Meniul Open apelează funcția *imgPreview()* ce deschide o fereastră de dialog în care se alege path-ul unui fișier, iar dacă acesta reprezintă o imagine validă ce poate fi citită de pachetul *OpenCV*, se creează matricea internă *pixelMatrix* ce va fi manipulată pe parcursul programului. De asemenea, se inițiază parametrii generali legați de modul de lucru cu canvas-ul și se mapează comenzile de bază pe apăsarea și ridicarea mâinii de pe butonul Left Click. După terminarea task-urilor din funcția *imgPreview(),* este apelată funcția *ExportImagePreview()* ce interpolează dimensiunile pozei originale, creând o miniatură a acesteia ce poate fi expusă în canvas, păstrând aspect ratio-ul imaginii.

Meniul New Canvas apelează funcția *newCanvas()* unde se creează o fereastră de dialog în care utilizatorul poate selecta înălțimea și lățimea unui nou canvas gol. Dacă înălțimea și lățimea sunt numere pozitive, atunci se creează un array cu elementele 255 în *pixelMatrix* folosind pachetul *numpy,* se inițializează parametrii generali legați de modul de lucru și se afișează asemănător noua imagine generată folosind funcția *ExportImagePreview().*

Meniul Save apelează funcția *saveImg()* în care se testează dacă există o imagine validă în pixelMatrix folosind un flag extern. Dacă imaginea este validă, se creează o fereastră de dialog unde se poate selecta un factor în procente de la 20% la 100% pentru rata de compresie JPEG. Valoarea inițială a acestui factor este 95%, deoarece acesta produce cele mai mari reduceri ale dimensiunii imaginii fără a pierde detalii importante. După apăsarea butonului Proceed, factorul este înregistrat, se deschide o fereastră de dialog în care utilizatorul poate selecta path-ul unde va salva imaginea cu factorul de compresie ales.

Meniul Efecte Statice apelează funcția *windowEfecte().* Dacă imaginea citită sau creată este validă, atunci se creează mai multe butoane din tkinter ce vor deservi la aplicarea efectelor statice, fiecare cu o funcție asociată.

Vignette – Se creează două nuclee pe axele Ox și Oy de dimensiuni relative la matricea *pixelMatrix* care centrează valorile (filtre Gaussiene). Aceste două nuclee se înmulțesc și creează o mască ce se înmulțește cu 255 (valoarea maximă pentru un pixel) rezultând o mască finală ce poate fi aplicată fiecărei culori pentru a centra luminozitate.

Edge detection – Implementat prin functia Canny din pachetul opencv. Imaginii i se reduce zgomotul pentru ca diferentele bruste de culoare pot genera margini nedorite. Acest proces se face folosind tot un filtru Gaussian a carui ecuatie poate fi vazuta în *[Fig.5.1.1.*]. Se folosesc doua filtre Sobel pe ambele axe ce sunt transpuse în coordonate polare ca în *[Fig.5.1.2.]*. În continuare se doreste reducerea pixelilor intercalati prin alegerea unor valori limita pentru unghi și magintudine, rezultand magnitudinea finala ce va reprezenta rezultatul filtrului Canny.

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Fig.5.1.2. Obtinerea filtrului Gaussian

$$|G| = \sqrt{I_x^2 + I_y^2},$$
$$\theta(x, y) = arctan\left(\frac{I_y}{I_x}\right)$$

Fig.5.1.2. Transformarea în coordonate polare

Grayscale – Se trece la spațiul gri făcând media celor 3 culori și stocându-le pe toate cele 3 canale (*OpenCV* interpretează un singur canal, cunoscând proprietatea *cv2.COLOR_GRAY*).

Vertical Flip – Se consideră pivot jumătatea înălțimii. Toți pixelii deasupra pivotului vor fi schimbați cu toți pixelii de sub pivot, rezultând în imaginea oglindită față de axa Oy. *Pixel[i][j] <=> pixel[înălțime-i][j], i=0...pivot, j=0...lățime.*

Horizontal Flip – Se consideră pivot jumătatea înălțimii. Toți pixelii deasupra pivotului vor fi schimbați cu toți pixelii de sub pivot, rezultând în imaginea oglindită față de axa Oy. *Pixel[i][j] <=> pixel[i][lățime-j], j=0...pivot, i=0...înălțime.*

Sobel X – Se aplică un nucleu de convoluție *[Fig.5.1.3.]* ce afectează pixelii pe axa Ox, afectând diferențele bruște între culori, rezultând într-un filtru de bază pentru edge detection ce poate fi utilizat mai departe în combinație cu alte filtre.

Sobel Y – Se aplică un nucleu de convoluție *[Fig.5.1.3.]* ce afectează pixelii pe axa Oy, afectând diferențele bruște între culori, rezultând într-un filtru de bază pentru edge detection ce poate fi utilizat mai departe în combinație cu alte filtre.

$$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$
$$K_x \qquad\qquad K_y$$

Fig.5.1.3. K_x Sobel pe axa Ox, K_y Sobel pe axa Oy

Sobel Full – Se aplica cele doua filtre Sobel X și sobel Y pentru a crea o reprezentare mai exactă a marginilor unei imagini.

Efectele folosite pentru edge detection se aplică, de regula, pe imaginile transformate alb-negru.

Emboss – Se aplică un nucleu de convoluție *[Fig.5.1.4.]* ce scoate în evidență pixelii de pe diagonală, creând un efect de adâncime / reliefare a imaginii.

$$\begin{pmatrix} -3 & -2 & 0 \\ -2 & 1 & 2 \\ 0 & 2 & 3 \end{pmatrix}$$

Fig.5.1.4. Nucleu de convolutie pentru Emboss

Sepia – Aplicarea unui nucleu de transformare *[Fig.5.1.5.]* ce produce pixeli cu nuanțe din spectrul rosu-galben-maro, producând un efect asemănător colorizarii fotografiilor din trecut.

$$\begin{pmatrix} 0.393 & 0.769 & 0.189 \\ 0.349 & 0.686 & 0.168 \\ 0.272 & 0.534 & 0.131 \end{pmatrix}$$

Fig.5.1.5. Nucleu de transformare pentru Sepia

Remove BG – Se folosește de o gamă largă de filtre de edge detection și computer vision pentru a distinge subiectele imaginii de background. Tool-ul este gratuit și open source, dezvoltat de Kaleido, și poate fi accesat prin linie de comandă cu un API key generat pentru fiecare utilizator.

Crop – Se activează parametrul de cropping ce modifică funcționalitatea apăsării butonului click stânga în interiorul canvas-ului. La apăsarea butonului se înregistrează coordonatele din interiorul canvas-ului. În timpul cât este apăsat pe canvas se desenează un dreptunghi cu contur galben folosind funcția *create_rectangle* ce descrie zona pe care utilizatorul o selectează în momentul respectiv. La dezactivarea click-ului se înregistrează coordonatele finale și dreptunghiul folosit ca preview dispare. Coordonatele respective sunt apoi translatate la coordonatele reale ale imaginii, canvasul de preview și imaginea având dimensiuni diferite, și se extrage zona dreptunghiulară descrisă de X,Y inițiale și X,Y finale, creând o nouă matrice ce va fi afișată folosind funcția *ExportImagePreview().* Poza anterioară este salvată într-o matrice de revert ce va folosi la efectul Revert Fake Zoom.

Revert Fake Zoom – Datorită diferenței dintre dimensiunile canvas-ului și ale imaginii originale, respectiv a incompatibilității directe a pachetelor tkinter și OpenCV, zoom-ul imaginii se poate realiza folosind funcția Crop. Pe aceasta se pot aplica absolut orice efecte, iar

la apăsarea butonului Revert Fake Zoom, cadrul dreptunghiular modificat este suprapus la coordonatele salvate, deasupra matricei inițiale salvate la pasul Crop, astfel simulându-se efectul de zoom.

Undo – Implementarea butonului undo se folosește de o stivă în care se stochează imaginile după fiecare modificare făcută și la creare folosind *.append()*. La fiecare apăsare a butonului undo se reinițializează matricea cu *pixelMatrix = pixelMatrixStack.pop()*, ținând cont de ultima funcție apelată, parametru ce se actualizează la fiecare apel de funcție. Deoarece matricea actuală și matricea din Stivă sunt aceleași la prima apăsare a butonului undo, dacă ultima funcție apelată nu a fost undo, se va da pop în stivă de două ori pentru a ajunge la modificarea anterioară. Dacă ultima funcție a fost undo, atunci se va da pop o singură dată.

Efecte Dinamice – apelează funcția ce descrie efectele dinamice, în același mod ca și selectarea opțiunii din meniul File.

Meniul Efecte Dinamice deschide o fereastră ce modifică în mod dinamic parametrii unei imagini ținând cont de o intensitate a efectului, ce poate fi selectată folosind sliderele dedicate pentru fiecare efect.

Brightness – cu intensitate cuprinsă între 0 (imagine neagră) și 512 (imagine albă) ce este scalată la valori cuprinse între -255 și 255. Modifică luminozitatea imaginii.

Contrast – cu intensitate cuprinsă între 0 și 255 ce este scalată la valori cuprinse între -127 și 127. Modifică diferența de luminozitate / intensitate a culorilor din regiunile imaginii.

Funcțiile Brightness și Contrast sunt tratate într-o singură funcție ce face o sumă ponderată a imaginii cu brightness și a imaginii cu contrast, fiecare imagine fiind o sumă ponderată a imaginii inițiale cu un factor de transparență și a imaginii înmulțite cu factorul de intensitate.

Hue – cu intensitate cuprinsă între 0 și 100. Imaginea este transformată din spațiul RGB în spațiul HSV, iar la primul canal se adaugă intensitatea selectată resetând valoarea după pragul 180. Acest efect descrie nuanța pură a unei culori de bază.

Saturation – cu o intensitate cuprinsă între 0 și 200, transformate în valori cuprinse între 0 și 2, 0 fiind o imagine nesaturată și 2 suprasaturată. În aceeași funcție ca modificarea Hue, imaginea este transformată în spațiul HSV și pe al doilea canal se înmulțește cu această intensitate păstrând valorile în gamă 0, 255.

Blur Mod – Se folosește un *nucleu Gaussian de Blur* de dimensiune *2*BlurMod+1 [Fig.5.1.6.]*, pentru a netezi imaginea și a ignora trecerile bruște dintre culori.

$$\frac{1}{273}\begin{pmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{pmatrix}$$

Fig.5.1.6. Nucleu Gaussian de Blur de dimensiune 5x5 (BlurMod = 2)

Blur Sigma – Deviația standard pe axa Ox a nucleului de convoluție. Estompează conturul pe axa Ox, parametru folosit în crearea *nucleului Gaussian de Blur* în combinație cu Blur Mod. Ambii parametri variază între 0 și 30.

Sharpness – Variază între 0 și 60. Se bazeaza pe un nucleu de convoluție *[Fig.5.1.7.]* ce se înmulțește cu intensitatea efectului scalată la intervalul 0, 2 urmărind apoi o corecție a valorii centrale din nucleu. Se obține ascuțimea imaginii prin intensificarea pixelilor individuali, raportati la cei vecini.

$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{pmatrix} \cdot \frac{sharpness}{30}$$

Fig.5.1.7. Nucleu de Sharpness

Solarization – Variază între 0 și 255, factor de intensitate ce reprezintă pragul de culoare la care culoarea se inversează. Pentru fiecare canal de culoare se verifică dacă valoarea pixelului depășește acest prag, caz în care se scade din 255 și se atribuie pixelului din canalul respectiv.

Efectele din meniul Efecte Dinamice pot fi combinate între ele, dar și cu efectele din Efecte Statice. Efectele dinamice se salvează doar la închiderea ferestrei „Efecte Dinamice”, moment în care imaginea rezultată se adaugă la stiva Undo.

Meniul Painting – Deschide o nouă fereastră ce prezintă un tool de desenat. Checkbox-ul „Painting Brush” activează modul de desenare ce diferențiază între tool-ul de paint și un tool ce va fi dezvoltat în viitor. Dropdown options prezintă 3 brush-uri: Default în care se desenează

o linie folosind funcțiile createLine între două seturi de coordonate succesive înregistrate, de grosime Brush Size; brush-ul Star creează o multitudine de linii la un unghi generat aleator ce aparține intervalului *[0, Star Angle]* (echivalent *[-Star Angle/2, Star Angle/2],* obținând punctele pe axa x cu *cos()* și axa Oy cu *sin().* Aceste linii sunt de lungime Brush Size și numărul lor este dat de parametrul Density; brush-ul Spray creează o multitudine de puncte pe o rază dată de Brush Size, cu o densitate dată de Density. La selectarea unei opțiuni din dropdown se setează funcția asociată fiecărui brush pe butonul <*B1-Motion*>.

Meniul Reset aduce imaginea în momentul inițial, cel al deschiderii prin *Open* sau cel al creării canvas-ului prin opțiunea *New Canvas*.

Meniul Resize deschide o fereastră ce așteaptă un factor de scalare ca input de la utilizator prin intermediul unui slider și afișează noua rezoluție a imaginii în urma aplicării acestui factor. La apăsarea butonului *Resize* din cadrul ferestrei se aplică o scalare din pachetul skimage cu factorul de scalare introdus de utilizator prin slider.

Toate modificările: *Efecte Slider, Efecte Dinamice* și *Painting* sunt compatibile între ele, interacționând corect cu butonul de *Undo, Crop* și *Revert Fake Zoom*.

Meniul Custom – conține cel puțin 3 submeniuri : *Custom Kernel, Add Functions, Modify Functions.*

Meniul Custom Kernel – Permite utilizatorului să aplice nuclee de convoluție personale, de dimensiune *3x3, 5x5* sau *7x7.* Utilizatorul introduce valorile individuale în casetele de text, putând introduce și expresii matematice de tipul „*7/2*5*” ce vor fi evaluate ulterior. Utilizatorul poate naviga prin casetele de text folosind tastele *: Up, Down, Left, Right.*

Meniul Add Functions – Deschide o nouă fereastra cu un *text-input* care permite utiliozatorului să își creeze propriile funcții în Python ce vor primi ca parametru matricea de pixeli și vor returna matricea de pixeli alterată în urma funcției ce a fost scrisă de utilizator. Acesta va specifica numele funcției în primul comentariu din fișier.

Meniul Modify Functions – Se deschide o nouă fereastră cu o listă de selecție în care se află funcțiile generate de către utilizator. Acesta poate apăsa butonul *Delete* pentru a șterge funcția respectivă sau *Modify* pentru a deschide *text-input-ul* și a modifica funcția respectivă.

Funcțiile adăugate/sterse se vor adăuga ca și submeniuri în meniul *Custom.*

5.2. Modulul de recunoastere a mâinii și control al cursorului

Modulul de recunoaștere este alcătuit din două fișiere: un fișier de bază numit *HandDetectionModule.py* ce se folosește de funcțiile din pachetul *mediapipe* pentru a recunoaște elementele de bază necesare procesării unei astfel de aplicații și fișierul aplicație numit *VirtualMouse.py* ce interpretează rezultatele procesate de obiectul creat prin *HandDetectionModule* și realizează cerința dorită: controlul cursorului. Aplicația are la bază biblioteca mediapipe dezvoltată de *Google*, ce oferă o multitudine de facilități din domeniul computer vision în detectarea și standardizarea recunoașterii corpului uman.

HandDetectionModule implementează o clasă numită *HandDetector* ce are ca și inițializare pregătirea variabilelor cum ar fi numărul de mâini pe care se așteaptă să le găsească într-o imagine, id-urile punctelor de interes din mână, parametrii ce țin de credibilitatea pe care trebuie să o aibă algoritmul pentru a considera că ceea ce descoperă este o mână. Pe lângă inițializare se mai întâlnesc și:

findHands(): funcție ce găsește poziția mâinii dintr-o imagine și salvează această poziție. Funcția are posibilitatea de a desena pe imaginea trimisă acest rezultat pentru o exemplificare vizuală a funcționalității.

findPosition(): funcție ce găsește poziția punctelor de interes din imagine (articulațiile degetelor, vârfurile degetelor, forma palmei) și desenează, în caz că parametrul pentru Draw este setat pe true, puncte în zonele articulațiilor și un chenar dreptunghiular în jurul mâinii.

fingersUp(): funcție ce compară poziția vârfurilor degetelor raportată la poziția articulației inferioare, determinând astfel care deget este ridicat. Această funcție returnează o listă de 5 elemente binare, ce reprezintă poziția fiecărui deget: 1 – ridicat, 0 – coborât.

findDistance(): funcție ce calculează și returnează distanța dintre 2 degete, ilustrând pe imaginea afișată o linie între acestea și puncte care determină unde a fost desenată linia.

VirtualMouse implementează funcționalitatea pe baza informațiilor generate de modulul *HandDetectionModule*. Acesta primește prin intermediul bibliotecii *OpenCV* capturi de la camera web. Se creează obiectul *HandDetector* cu *maxhands=1* (obiectul aplicației nu presupune două mâini). În interiorul unei bucle infinite se citește imagine din captura camerei web, pe baza căreia se obțin informațiile necesare prin apelarea funcțiilor *findHands(),*

findPosition() și *fingersUp()*. Deoarece camera și ecranul au dimensiuni diferite, și aici se face o scalare asemănătoare ca la interacțiunea dintre canvas și imaginea originală. Se verifică în instrucțiuni alternative separate dacă avem degetele necesare unei anumite acțiuni ridicate și anume:

Pentru degetul arătător ridicat dorim să mișcăm cursorul pe ecran. Astfel vom calcula coordonatele corespunzătoare pe ecran pe baza poziției pe care degetul arătător o are în imaginea obținută din webcam, aplicăm un factor de smoothening pentru a compensa zgomotul dat de calitatea scăzută a camerei web, mișcările / tremuratul mâinii și factorul de credibilitate oferit modulului. După calcularea coordonatelor reale vom mișca cursorul folosind biblioteca autopy, ținând cont că imaginea de la camera web este în oglindă.

Pentru degetul arătător și cel mijlociu ridicate vom calcula distanța dintre aceste două folosind funcția findDistance(), iar dacă aceasta este mai mică decât un prag setat vom apăsa click stânga folosind modulul *autopy.mouse.click()*.

Pentru degetul arătător și cel mic ridicate vom mișca cursorul exact ca în cazul normal, doar că vom ține click stânga apăsat folosind *autopy.mouse.toggle()*. Când această condiție nu este îndeplinită, se va ridica folosind aceeași funcție pentru a putea muta cursorul fără a ține click apăsat.

Din motive de testare a performanței, pe lângă toate chenarele descriptive funcționalității, există și un *FPS (Frames per second)* counter în care putem observa cum se schimbă viteza de procesare a imaginilor prin testarea și validarea diferitelor poziții ale degetelor, afișând la fiecare iterație această imagine.

Modulul poate fi lansat din meniul *Extra* → *Start Mouse*. La apăsarea acestui buton se lansează aplicația *VirtualMouse.py* pe un thread nou. Aplicația așteaptă ca primul frame din înregistrarea camerei web să conțină o mână pentru a confirma faptul că nu a fost lansată din greșeală. Pentru închiderea aplicației *VirtualMouse* se poate accesa *Stop Mouse* din meniul *Extra* sau prin închiderea ferestrei principale root.

6. Concluzii, planuri de viitor și limitari

O astfel de aplicație poate oferi utilizatorilor ce își doresc o transparență a codului și o implementare a mai multor efecte decât cele comerciale disponibile, cu posibilitatea de a-și introduce efecte proprii prin modificarea codului, dar și posibilitatea persoanelor ce nu pot manipula un mouse în mod direct să beneficieze de editarea imaginilor la computer.

Din motive ce țin de dependența pachetelor, aplicația este realizată în *Python 3.7,* dar poate fi migrată la versiunea *3.11*. Versiunea *3.7* s-a dovedit a fi una care a satisfăcut performanța manipulării imaginilor prin diverse biblioteci, dar prin dezvoltarea unei biblioteci dedicate ce va face transformările necesare într-un limbaj de nivel mai jos *(C / C++)* se poate porta la versiunea *3.11*. Versiunea *3.11* este limita actuală dată de versionarea pachetului *mediapipe*. Acesta poate fi înlocuit cu varianta de la *TensorFlow* pentru un suport mai dezvoltat din partea comunității din viitor.

În ceea ce privește implementarea de noi funcționalități în modulul de editare imagine și desen, obiectivele principale în prezent sunt: introducerea manipulării fișierelor *PNG* în adevăratul sens al cuvântului, implementarea funcționalităților de tip *layer*, optimizarea funcției *undo(),* implementarea de efecte noi și o modalitate de a introduce interactiv cod în aplicație pentru a introduce efecte customizate, utilizarea unui brush pentru a aplica acele efecte în locuri izolate, implementarea unei soluții eficiente pentru *zoom & pan*, dezvoltarea unei biblioteci de legătură între *tkinter* și restul bibliotecilor de manipulare a imaginilor, o posibilă migrație către *PyQT* sau *wxPython* și introducerea mai multor tool-uri bazate pe machine learning pentru a oferi o experiență completă.

Referitor la obiectivele pentru îmbunătățirea modulului de recunoaștere a mâinii: implementarea diverselor gesturi și semne pentru a accesa meniuri în mod direct, preprocesarea imaginii pentru a asigura utilizatorul că fundalul sau mediul din care lucrează nu îi va afecta performanța aplicației.

Limitările actuale ale dezvoltării aplicației sunt legate de disponibilitatea hardware-ului (puterea de procesare, calitatea și frame-rate-ul camerei web), mediul exterior (luminozitatea și unghiul de poziționare în fața camerei), dar și de soluțiile software disponibile momentan (biblioteca *mediapipe*)

7. Bibliografie

1. Documentatia oficiala a bibliotecii mediapipe:
https://mediapipe.readthedocs.io/en/latest/solutions/hands.html

2. Documentatia oficiala a bibliotecii pillow (PIL):
https://pillow.readthedocs.io/en/stable/

3. Documentatia oficiala a bibliotecii tkinter
https://docs.python.org/3/library/tk.html

4. Documentatia oficiala a bibliotecii opencv
https://docs.opencv.org/4.5.4/

5. Imaginea folosita pentru testare:
https://pixabay.com/photos/pug-dog-sunglasses-pink-glasses-8632718/

8. Anexa

Nolan Arbaugh Speech:
https://www.youtube.com/watch?v=79VvxBStbWY&t=1473s

Neuralink Live Update - March 2024:
https://www.youtube.com/watch?v=ZzNHxC96rDE

2006 – Moving cursor though a brain implant:
https://www.nytimes.com/2006/07/13/science/13brain.html

Codul sursa – Disponibil incepand cu 09.05.2024:
https://github.com/zuch3e/Image-Editing-With-Hand-Controlled-Cursor