smart AInnovators

Multiplier Event in Poland

09/06/2024

Lead partner: Społeczna Akademia Nauk

smartAInnovators: Promoting AI-driven digital Transformation and innovation in VET Schools for social change and better Skills match with the labour market

Project number: 2021-1-PL01-KA220-VET-000033011

This project has been co-funded with support from the European Commission.
This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by the European Union

This document may be copied, reproduced or modified according to the above rules.
In addition, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.
All rights reserved. © Copyright 2021 smartAInnovators

CC BY NC SA

CONTEXT

Grant agreement	2021-1-PL01-KA220-VET-000033011
Programme	Erasmus+
Key action	Cooperation for innovation and the exchange of good practices
Action	Strategic Partnerships
Project acronym	smartAInnovators
Project title	smartAInnovators: Promoting AI-driven digital Transformation and innovation in VET Schools for social change and better Skills match with the labour market
Project starting date	01/01/2022
Project duration	30 months
Project end date	01/07/2024

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Contents

Description of the Event 4

Photos/Screenshots 5

Dissemination Material 8

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

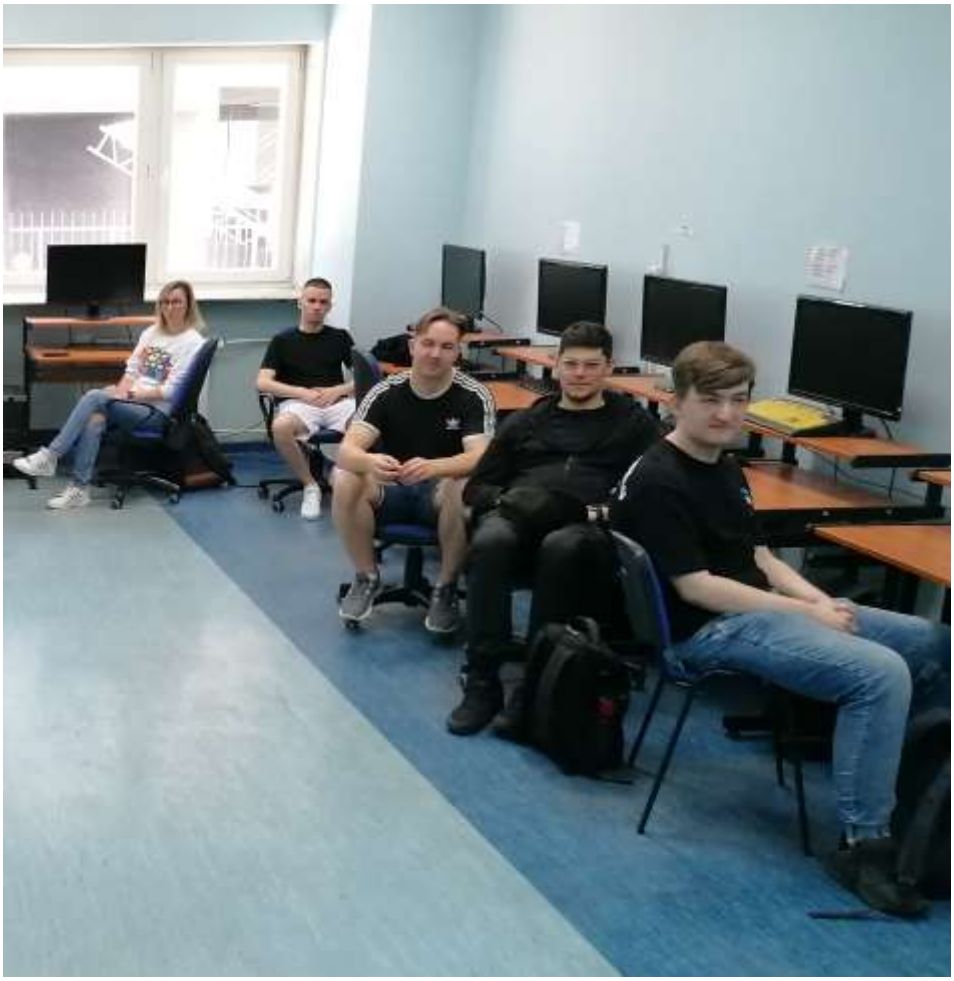
Description of the Event

Date(s)	09.06.2024
Place/City	*Warsaw*
Coordinator name and email	*Paweł Pełczyński, ppelczynski@san.edu.pl*
Number of participants	*28*
Agenda	1. *Presentation of the project objectives* 2. *Discussion of the workshop program for VET students* 3. *Presentation of selected project results* 4. *Workshops on training AI to resolve object recognition tasks in images using the LOBE program* 5. *Workshops on using AI methods in own applications developed in Python*
Description of activities	• ***Presentation of facial recognition of the workshop participant*** *This activity began with the recording of several shots of the faces of a group of several people with a USB camera. Then the AI system was trained to perform facial recognition tasks. After training the model, the system was tested, i.e. recognizing who is currently in front of the camera. Workshop participants had the opportunity to assess the effectiveness of recognition, the system's reaction to the absence of a face or the appearance of another person.* • ***Presentation of object recognition from the environment*** *The activity began with recording a dozen or so shots of selected objects with a USB camera, e.g. a computer keyboard, monitor, pen on the desk, etc. Then, workshop participants started teaching the AI system to recognize these objects. The trained model was tested. Object recognition that the AI system learned followed. It was possible to assess the effectiveness of recognition, observe the system's reaction to the absence of an object or the appearance of an object other than the object from the training set.* 1. ***Presentation of using face tracking model to create simple games*** *Workshop participants observed the effectiveness of face tracking depending on the distance from the camera, the number of people in the camera's field of view, and lighting conditions. They also had the opportunity to see the use of a face tracking library in a game developed by students participating in the project, consisting in bouncing a virtual ball with their "head". Workshop participants experimented by introducing their own modifications to the program parameters and observing their impact on the quality of the game. The effects of distance from the camera, lighting conditions, and interference were observed due to the appearance of more people in the camera's field of view.*

Co-funded by the European Union

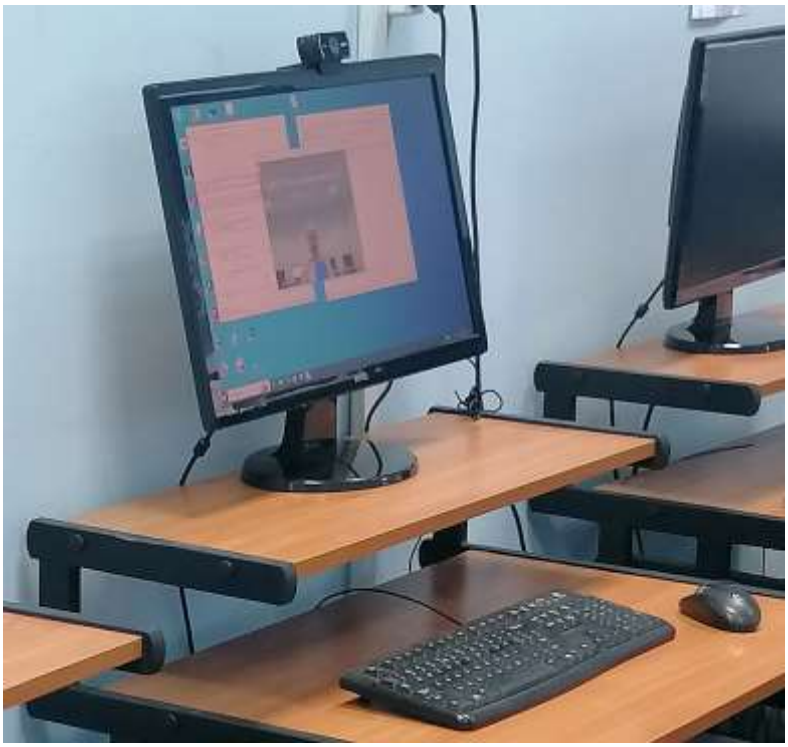
This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Photos/Screenshots



Co-funded by
the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



Co-funded by
the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by
the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Dissemination Material

Selected slides of project materials presented during the workshop:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Co-funded by
the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Source code of one of the games developed by VET students, presented to workshop participants:

```
# Odbijanie pilki na ekranie
import numpy as np
import cv2

#####################################################################
# Zmienne globalne - parametry lokalizacji i prędkości
# paletki oraz piłeczki

# Współrzędne środka głowy
head_x = 300
head_y = 200
# Wektor ruchu głowy (na ramke)
head_u = 0
head_v = 0

# Współrzędne środka piłeczki
ball_x = 300
ball_y = 30
# Wektor ruchu piłeczki (na ramke)
ball_u = 0
ball_v = 0
# Przyrost predkosci
dv = 2
# Promień okręgu - piłeczki
ball_radius = 50
# Kolor piłeczki BGR
ball_color = (255, 100, 50)

# Wynik
Liczba_odbic = 0

# Pomniejszenie obrazu do analizy
imreduce = 1


#####################################################################

def detectAndDisplay(big_frame):

    global ball_x, ball_y
    ball_center = (ball_x,ball_y)
    global ball_u, ball_v
    face_detected = 0
    global Liczba_odbic
    global head_x
    global head_y
    global head_u
    global head_v

    # Pomniejszanie obrazu 2x
    height, width = big_frame.shape[:2]
    new_size = (width // imreduce, height // imreduce)
    small_frame = cv2.resize(big_frame, new_size, interpolation=cv2.INTER_AREA)
    height, width = small_frame.shape[:2]

    # Flip the image horizontally
    frame = cv2.flip(small_frame, 1)

    # Rozdzielenie obrazu na składowe BGR
    Blue, Green, Red = cv2.split(frame)

    # Konwertowanie obrazu do skali szarości
    frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Wyrównanie histogramu dla lepszej jakości obrazu
    frame_gray = cv2.equalizeHist(frame_gray)

    # Wykrywanie twarzy
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
faces = face_cascade.detectMultiScale(frame_gray)

for (x,y,w,h) in faces:
    # Znalezienie środka twarzy
    center = (x + w//2, y + h//2)

    # Obszar zainteresowania (Region of Interest - ROI) - obszar twarzy
    faceROI = frame_gray[y:y+h, x:x+w]

    # Wykrywanie oczu w obszarze twarzy
    eyes = eyes_cascade.detectMultiScale(faceROI)

    # Opcjonalnym warunkiem obecności gracza jest wykrycie twarzy i co najmniej jednego oka
    if len(eyes)>0:
    # if 1:

        face_x = x + w//2
        face_y = y + h//2
        head_u = face_x - head_x
        head_v = face_y - head_y
        head_x = face_x
        head_y = face_y
        frame = cv2.ellipse(frame, center,(w//2, h//2), 0, 0, 360, (255, 0, 255), 4)
        face_detected = 1
    else:
        head_u = 0
        head_v = 0

    for (ex, ey, ew, eh) in eyes:
        # Znalezienie środka oka i narysowanie okręgu wokół niego
        eye_center = (x + ex + ew//2, y + ey + eh//2)
        radius = int((ew + eh) * 0.25)
        frame = cv2.circle(frame, eye_center, radius, (255, 0, 0), 4)


# Rysowanie pileczki
frame = cv2.circle(frame, ball_center, ball_radius, ball_color, cv2.FILLED)

# Aktualizacja parametrów lokalizacji, orientacji i ruchu paletki
# Aktualizacja parametrów lokalizacji i ruchu oraz piłeczki
ball_x = ball_x+ball_u
ball_y = ball_y+ball_v
ball_u = ball_u
ball_v = ball_v+dv

# Obsluga zderzenia
if face_detected:

    dist = np.sqrt((face_x-ball_x)**2+(face_y-ball_y)**2)
    mindist = ball_radius+np.sqrt((w//2)**2+(h//2)**2)

    # Jeśli dystans jest poniżej minimalnego i:
    # 1 opcja: piłka powyżej głowy
    #if dist<mindist and ball_y<face_y:
    # 2 opcja: piłka leci w dół
    if dist<mindist and ball_v>0:
    # to kolizja: obliczenie zgodnie ze sprężystmy odbiciem kulki
    # od nieruchomej kulki


        #Wektor normalny:
        n_x = (ball_x-face_x)/dist
        n_y = (ball_y-face_y)/dist
        #Wektor tangencjalny:
        tan_x = - n_y
        tan_y = n_x

        ball_u = -(ball_u*n_x+ball_v*n_y)*n_x+(ball_u*(-n_y)+ball_v*n_x)*(-n_y)
        ball_v = -(ball_u*n_x+ball_v*n_y)*n_y+(ball_u*(-n_y)+ball_y*n_x)*n_x
        #Tlumienie wskutek nieidealnego odbicia i dodanie wektora ruchu głowy:
        #ball_u = int((ball_u+head_u) * 0.9)
```

Co-funded by the European Union

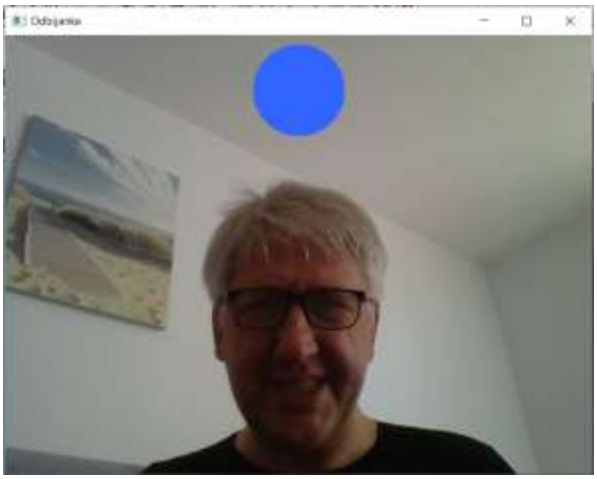
This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

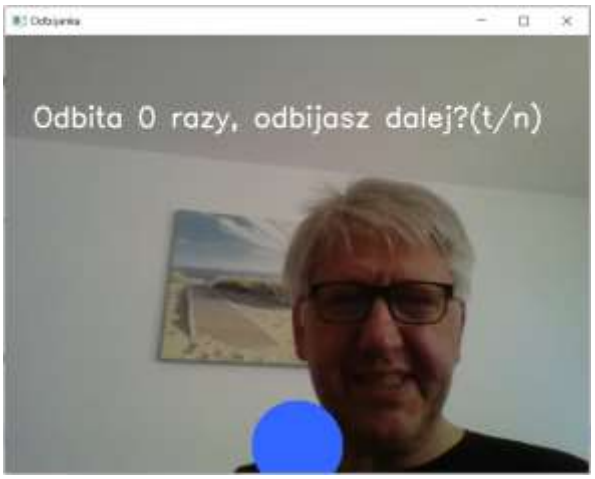
```
            ball_u = int((ball_u) * 0.5)
            ball_v = int((ball_v+head_v) * 0.9)

            Liczba_odbic = Liczba_odbic + 1

    # Test pozostania piłki w grze
    if ball_x>0 and ball_x<width and ball_y>0 and ball_y<height:
        # Wyświetlenie obrazu
        cv2.imshow("Odbijanka", frame)
        return 1
    else:
        # Koniec rozgrywki, czy kolejna?
        text = "Odbita "+str(Liczba_odbic)+" razy, odbijasz dalej?(t/n)"
        org = (30, 100)
        fontFace = cv2.FONT_HERSHEY_SIMPLEX
        fontScale = 1
        color = (255, 255, 255)  # Biały
        thickness = 2
        # Informacja
        frame = cv2.putText(frame, text, org, fontFace, fontScale, color, thickness)

        # Wyświetlenie obrazu
        cv2.imshow("Odbijanka", frame)

        # Czekaj na naciśnięcie klawisza
        key = cv2.waitKey(0)
        # Sprawdź, czy naciśnięty klawisz to 't'
        if key == ord('t') or key == ord('T'):
            ball_x = width//2
            ball_y = 30
            ball_u = 0
            ball_v = 0
            Liczba_odbic = 0

            return 1
        else:
            return 0


######################################################################
# Program główny
######################################################################

# Ładowanie klasyfikatorów kaskadowych do wykrywania twarzy i oczu
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_alt.xml')
eyes_cascade            =            cv2.CascadeClassifier(cv2.data.haarcascades            +
'haarcascade_eye_tree_eyeglasses.xml')

# Otwarcie strumienia wideo z kamery
camera_device = 0
capture = cv2.VideoCapture(camera_device)
if not capture.isOpened():
    print("--(!) Błąd podczas otwierania strumienia wideo")
    exit()

while True:
    ret, im_frame = capture.read()
    if im_frame is None:
        print("--(!) Brak klatki wideo -- Przerwanie!")
        break

    if not detectAndDisplay(im_frame):
            break


    if cv2.waitKey(10) == 27:  # Naciśnięcie klawisza 'Esc' kończy program
        break

# Zamknięcie okien
cv2.destroyAllWindows()
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Examples of how the game works:





Co-funded by
the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

www.smartainnovators.eu