VirtualEye - Life Guard For Swimming Pools To Detect Active Drowning Using IBM Cloud

1. INTRODUCTION

1.1 Overview

Swimming is one of the best exercises that helps people to reduce stress in this urban lifestyle. Swimming pools are found larger in number in hotels, and weekend tourist spots and barely people have them in their house backyard. Beginners, especially, often feel it difficult to breathe underwater which causes breathing trouble which in turn causes a drowning accident. Worldwide, drowning produces a higher rate of mortality without causing injury to children. Children under six of their age are found to be suffering the highest drowning mortality rates worldwide. Such kinds of deaths account for the third cause of unplanned death globally, with about 1.2 million cases yearly.

1.2 Purpose

To overcome this conflict, a meticulous system is to be implemented along the swimming pools to save human life. By studying body movement patterns and connecting cameras to artificial intelligence (AI) systems we can devise an underwater pool safety system that reduces the risk of drowning. Usually, such systems can be developed by installing more than 16 cameras underwater and ceiling and analyzing the video feeds to detect any anomalies. but AS a POC we make use of one camera that streams the video underwater and analyses the position of swimmers to assess the probability of drowning, if it is higher then an alert will be generated to attract lifeguards' attention.

2. LITERATURE SURVEY

2.1 Existing problem

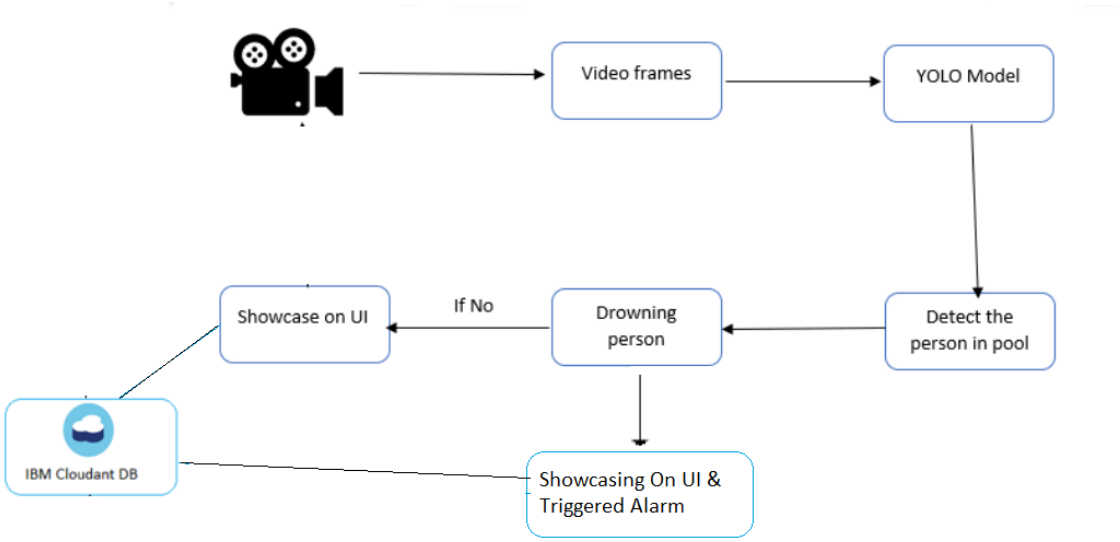
The existing problem of swimming is one of the best exercises that helps people to reduce stress in this urban lifestyle. Swimming pools are found larger in number in hotels, and weekend tourist spots and barely people have them in their house backyard. Beginners, especially, often feel it difficult to breathe underwater which causes breathing trouble which in turn causes a drowning accident. Worldwide, drowning produces a higher rate of mortality without causing injury to children. Children under six of their age are found to be suffering the highest drowning mortality rates worldwide. Such kinds of deaths account for the third cause of unplanned death globally, with about 1.2 million cases yearly.

2.2 Proposed solution

The proposed solution is to overcome this conflict, a meticulous system is to be implemented along the swimming pools to save human life. By studying body movement patterns and connecting cameras to artificial intelligence (AI) systems we can devise an underwater pool safety system that reduces the risk of drowning. Usually, such systems can be developed by installing more than 16 cameras underwater and ceiling and analyzing the video feeds to detect any anomalies. but AS a POC we make use of one camera that streams the video underwater and analyses the position of swimmers to assess the probability of drowning, if it is higher then an alert will be generated to attract lifeguards' attention.

3. THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software designing

Software Requirements:

- Cloudant
- Flask
- Numpy
- Opencv_python
- Playsound
- Progressbar33
- Requests
- Tensorflow

Hardware Requirements:

- Processor : Intel Core i3
- Hard Disk Space : Min 100 GB
- Ram : 4 GB
- Display : 14.1 “Color Monitor(LCD, CRT or LED

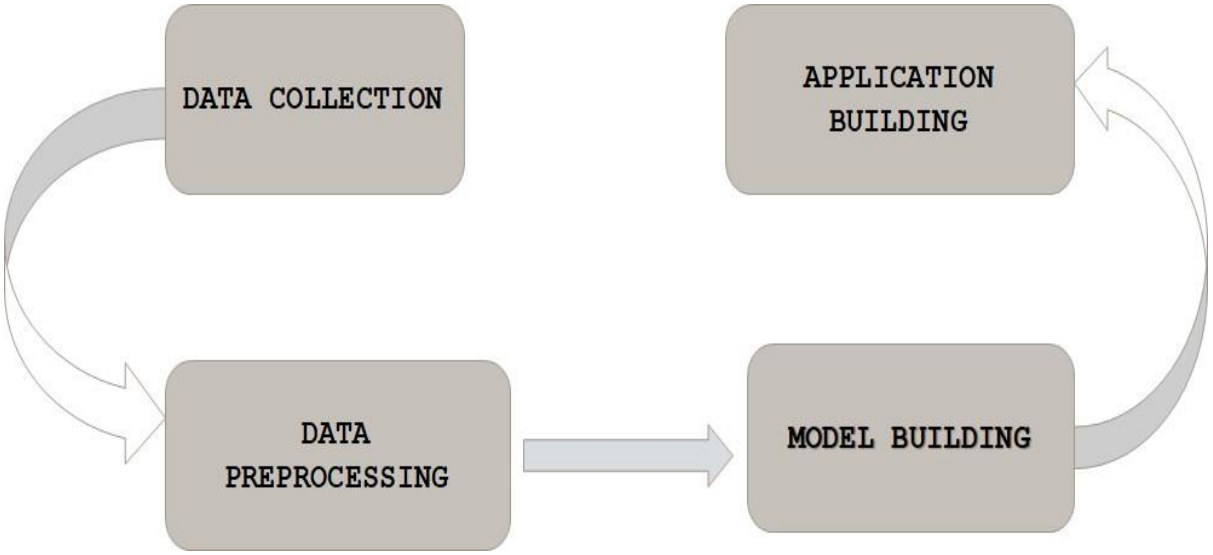
- Clock Speed : 1.67 GHz

4. EXPERIMENTAL INVESTIGATIONS

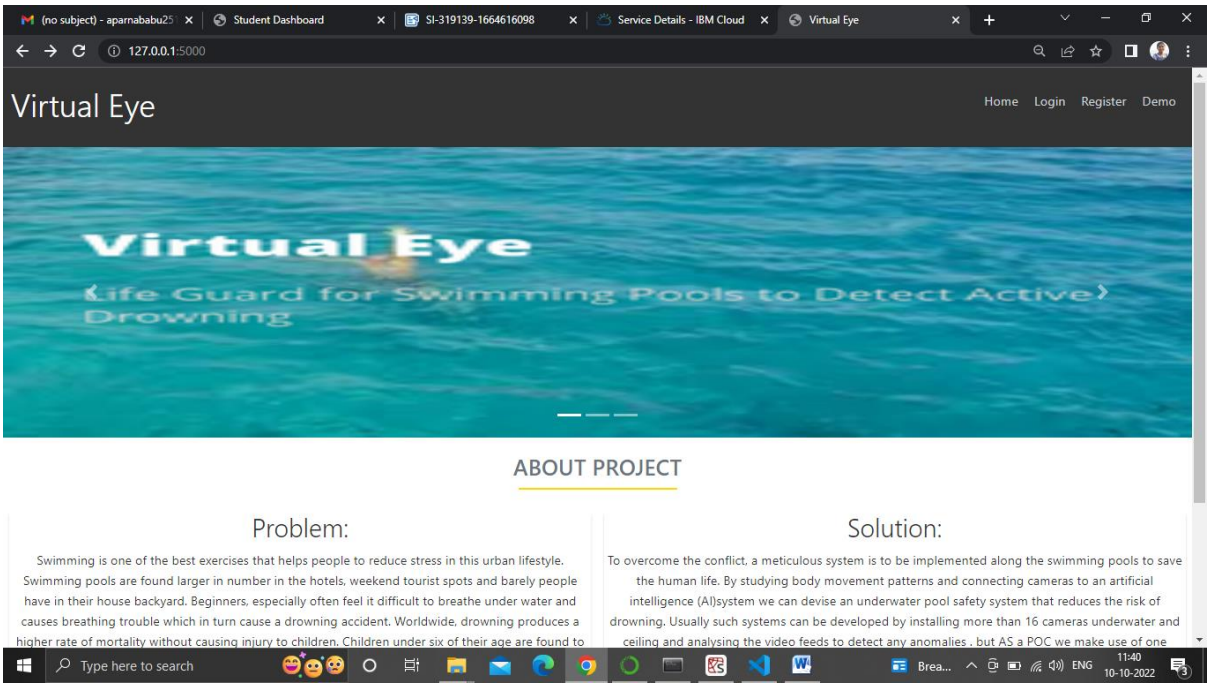
The studies show that it is to overcome this conflict, a meticulous system is to be implemented along the swimming pools to save human life. By studying body movement patterns and connecting cameras to artificial intelligence (AI) systems we can devise an underwater pool safety system that reduces the risk of drowning.

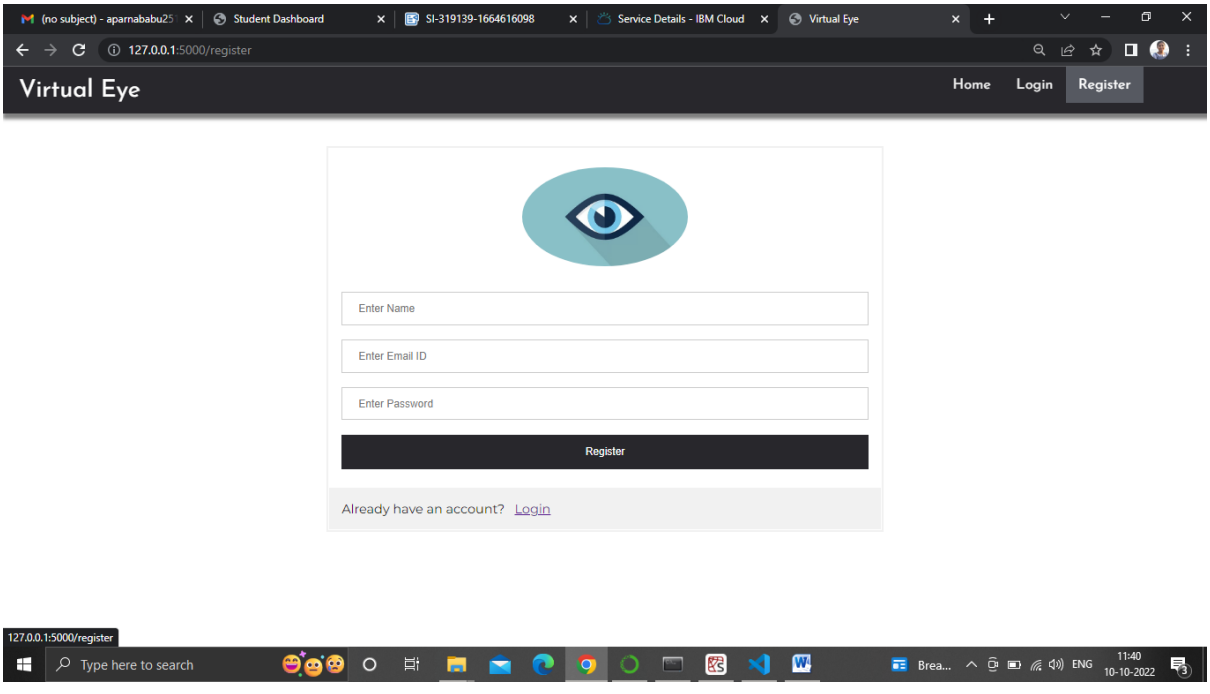
.

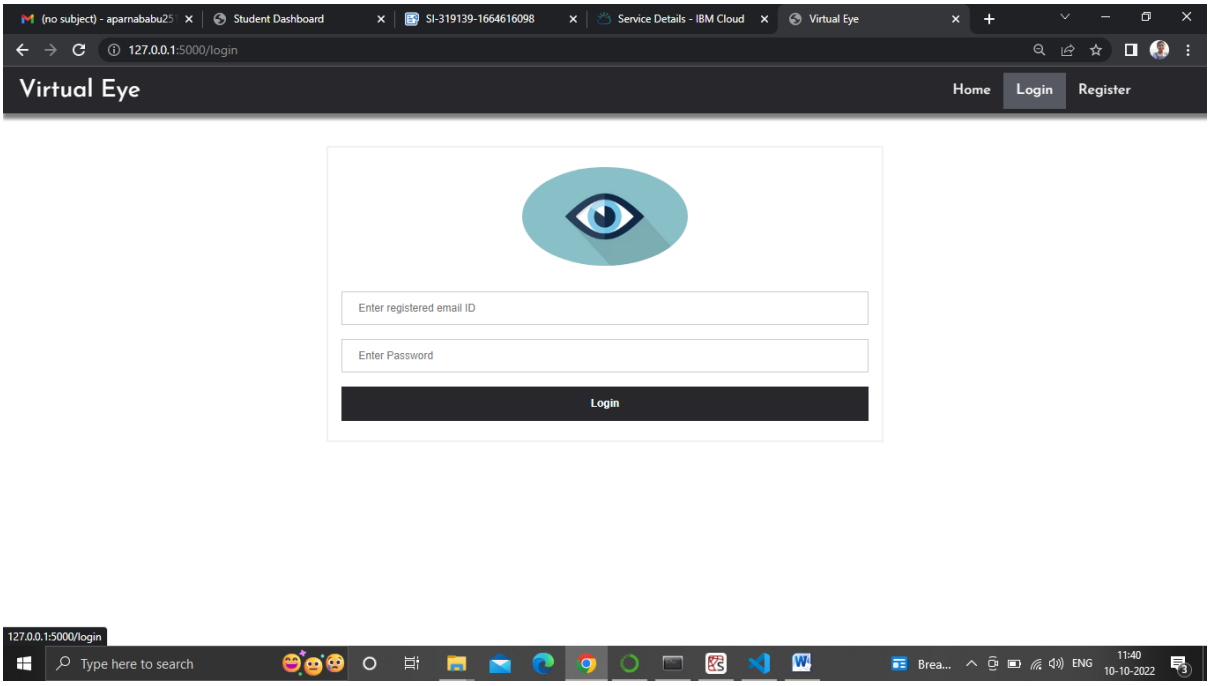
5. FLOWCHART



6. RESULT


(no subject) - aparnababu25
Student Dashboard
SI-319139-1664616098
Service Details - IBM Cloud
Virtual Eye
127.0.0.1:5000
Virtual Eye
Home Login Register Demo
Virtual Eye
Life Guard for Swimming Pools to Detect Active Drowning
ABOUT PROJECT
Problem:
Swimming is one of the best exercises that helps people to reduce stress in this urban lifestyle. Swimming pools are found larger in number in the hotels, weekend tourist spots and barely people have in their house backyard. Beginners, especially often feel it difficult to breathe under water and causes breathing trouble which in turn cause a drowning accident. Worldwide, drowning produces a higher rate of mortality without causing injury to children. Children under six of their age are found to
Solution:
To overcome the conflict, a meticulous system is to be implemented along the swimming pools to save the human life. By studying body movement patterns and connecting cameras to an artificial intelligence (AI)system we can devise an underwater pool safety system that reduces the risk of drowning. Usually such systems can be developed by installing more than 16 cameras underwater and ceiling and analysing the video feeds to detect any anomalies . but AS a POC we make use of one
Type here to search
ENG
11:40
10-10-2022


(no subject) - aparnababu25
Student Dashboard
SI-319139-1664616098
Service Details - IBM Cloud
Virtual Eye
127.0.0.1:5000/register
Virtual Eye
Home Login Register
Enter Name
Enter Email ID
Enter Password
Register
Already have an account? Login
127.0.0.1:5000/register
Type here to search
ENG
11:40
10-10-2022



7. ADVANTAGES & DISADVANTAGES

Advantages:

- It represents an additional level of safety and protection for swimmers.
- It can be installed in any type of pool and construction variant.
- It ensures effective and realiable drowning detection by limiting the number of alarms generated

Disadvantages:

- Data mining techniques does not help to provide effective decision making

8. APPLICATIONS

- Deep Learning technology is considered as one of the key technology used in virtual eye life guard for swimming pools to detect active drowning.

9. CONCLUSION

In this project, we provided a method to robust human tracking and semantic event detection within the context of video surveillance system capable of automatically detecting drowning incidents in swimming pool. Once we have the working drowning detection model we can feed live video footage of the swimming pool to it so that it can keep detecting continuously for any drowning activities. If drowning is detected it will be highlighted on the system screen as well as alarms will be raised to alert security guards so that they can initiate rescue

10. FUTURE SCOPE

Availability of better dataset, modern methodologies, and technologies with high computational power accompanied by high-quality surveillance cameras, will help to improve the accuracy of drowning detection & even can be used in adverse conditions.

After the implementation of all these essentials, this system also can be used on sea beaches for drowning detection

11. BIBILOGRAPHY

1. Wells, M.J., Virtual reality: technology, experience, assumptions, in Human Factors Soc. Bull. 1992, p. 1-3.

2. Slater, M. and M. Usoh, Body Centred Interaction in Immersive Virtual Environments, in Artificial Life and Virtual Reality, N.M. Thalmann and D. Thalmann (Eds). 1994, John Wiley and Sons: p. 125-148.

APPENDIX

Source Code

```
import re
import numpy as np
import os
from flask import Flask, app, request, render_template, redirect, url_for
from tensorflow.keras import models
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.python.ops.gen_array_ops import concat
import cvlib as cv
from cvlib.object_detection import draw_bbox
import cv2
import time
from playsound import playsound
import requests
```

```
app=Flask(__name__)
```

```
# Authenticate using an IAM API key
client = Cloudant.iam('e038c0a7-043a-4c4d-bedb-f8e0664d28cf-bluemix','Hs_dapTtf3_ldb4PXDMITLWJG5hzW6l8jYFoAFj3NDz-', connect=True)

# Create a database using an initialized client
my_database = client.create_database('my_database')
```

```
@app.route('/index.html')
def home():
    return render_template("index.html")
```

```
#registration page
@app.route('/register')
def register():
    return render_template('register.html')
```

```
#registration page
@app.route('/register')
def register():
    return render_template('register.html')

@app.route('/afterreg', methods=['POST'])
def afterreg():
    x = [x for x in request.form.values()]
    print(x)
    data = {
    '_id': x[1], # Setting _id is optional
    'name': x[0],
    'psw':x[2]
    }
    print(data)

    query = {'_id': {'$eq': data['_id']}}

    docs = my_database.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if(len(docs.all())==0):
        url = my_database.create_document(data)
        #response = requests.get(url)
        return render_template('register.html', pred="Registration Successful, please login using your details")
    else:
        return render_template('register.html', pred="You are already a member, please login using your details")
```

```python
#login page
@app.route('/login')
def login():
    return render_template('login.html')

@app.route('/afterlogin',methods=['POST'])
def afterlogin():
    user = request.form['_id']
    passw = request.form['psw']
    print(user,passw)

    query = {'_id': {'$eq': user}}

    docs = my_database.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if(len(docs.all())==0):
        return render_template('login.html', pred="The username is not found.")
    else:
        if((user==docs[0][0]['_id'] and passw==docs[0][0]['psw'])):
            return redirect(url_for('prediction'))
        else:
            print('Invalid User')
```

```python
@app.route('/prediction')
def prediction():
    return render_template('prediction.html')
```

```python
@app.route('/result',methods=["GET","POST"])
def res():
    webcam = cv2.VideoCapture('drowning.mp4')

    if not webcam.isOpened():
        print("Could not open webcam")
        exit()

    t0 = time.time() #gives time in seconds after 1970

    #variable dcount stands for how many seconds the person has been standing still for
    centre0 = np.zeros(2)
    isDrowning = False

    #this loop happens approximately every 1 second, so if a person doesn't move,
    #or moves very little for 10seconds, we can say they are drowning

    #loop through frames
    while webcam.isOpened():
        # read frame from webcam
        status, frame = webcam.read()

        if not status:
            print("Could not read frame")
            exit()
        # apply object detection
        bbox, label, conf = cv.detect_common_objects(frame)
        #simplifying for only 1 person
```

```
#s = (len(bbox), 2)
if(len(bbox)>0):
    bbox0 = bbox[0]
    #centre = np.zeros(s)
    centre = [0,0]
    #for i in range(0, len(bbox)):
        #centre[i] =[(bbox[i][0]+bbox[i][2])/2,(bbox[i][1]+bbox[i][3])/2 ]

    centre =[(bbox0[0]+bbox0[2])/2,(bbox0[1]+bbox0[3])/2 ]

    #make vertical and horizontal movement variables
    hmov = abs(centre[0]-centre0[0])
    vmov = abs(centre[1]-centre0[1])

    #there is still need to tweek the threshold
    #this threshold is for checking how much the centre has moved

    x=time.time()

    threshold = 10
    if(hmov>threshold or vmov>threshold):
        print(x-t0, 's')
        t0 = time.time()
        isDrowning = False

    else:
        print(x-t0, 's')
        if((time.time() - t0) > 10):
            isDrowning = True

    #print('bounding box: ', bbox, 'label: ' label ,'confidence: ' conf[0], 'centre: ', centre)
    #print(bbox,label ,conf, centre)
    print('bbox: ', bbox, 'centre:', centre, 'centre0:', centre0)
    print('Is he drowning: ', isDrowning)
```

```
    centre0 = centre
    # draw bounding box over detected objects

out = draw_bbox(frame, bbox, label, conf,isDrowning)

#print('Seconds since last epoch: ', time.time()-t0)
```