HOME AUTOMATIOIN: SMART FANS

PROJECT REPORT

HUMAN COMPUTER INTERACTION (CSE 4015)

Slot: B2+TB2

Name	Reg. no.
NAMAN DUTTA	17BCE2379
PRAJWOL DHUNGANA	18BCE2457

FACULTY: PROF. SWATHI J. N.

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Abstract

Today's generation is looking forward to easing everything in the best possible way. Human-Computer Interaction goal is to make system highly usable. In this project, we have come with an idea of Smart Fans. Objective is to use Smart Fans without any remote control. To achieve this, we have used gesture recognition. Using the least of resource (i.e., Camera Sensor), we can control the fan. Proposed system enables users to have interactions with Smart fans by recognizing user's gestures. Gestures will allow users to control speed of the fan and turn it on and off. Interacting with physical world using expressive body movements is much more comfortable and practical than just speaking. Main aim of project is to make system highly usable by all kind of people (including differently-abled people). Otsu thresholding is applied to separate foreground and background. Finally, template-based matching technique is developed using Principal Component Analysis (PCA) for recognition.

Objectives

- To recognize fingers in the environment.
    - Applying segmentation to divide the fingers and background.
    - Applying Otsu thresholding to obtain the fingers' outlines in image.
    - Applying high boost filter to sharpen the image.
- Counting the fingers so as to change the speed of the fan from 0 to 4
    - Cosine Angle Between the figures
    - Contours Counting
    - Event triggering

Literature Survey

S. N.	Author	Year	Publisher	Title	Description
1.	Mandeep Kaur Ahuja & Amardeep Singh Research Scholar and Professor Computer Engineering Department Punjabi University Main Campus, Patiala, INDIA mandeep.ahuja03@gmail.com	2015	IEEE	Static Vision Based Hand Gesture Recognition Using Principal Component Analysis	In this paper the hand gesture recognition system is developed using skin color model, Otsu Thresholding and PCA. the proposed model is not capable of working with the images containing hands of other than skin color. The proposed model does not evaluate the images clicked in other light colors where the hand gestures has been clicked and the model work only with static gesture .In future the system can be upgraded to support dynamic gestures and an application for controlling medical operations can be developed using the system.
2.	Hiroaki Yabe,Takuichi Nishimura,Ryuuichi Oka Tsukuba Research Center Tsukuba, 305 Japan yabe{nishi,oka}@tic.rucp.or.jp Toshiro Mukai	2015	IEEE	Recognition of gestures using morphological features of networks made of gesture motion images and word sequences	This paper proposes a method to recognize human gestures using both motion images and texts. The method uses two kinds of network models obtained from two kinds of sequences in a style of self-organization. They extract common and singular parts of gesture by analyzing the topology of the network of gesture motion images. The proposed method for recognizing gestures uses morphological features between two networks made of gesture motion images and word sequences. We showed the usefulness of the method through an experiment using

	Multimodal Functions Sharp Lab 1-9-2, Nakase, Mihama-ku, Chiba, Japan mukaiQiml.mkhar. sharp.co.jp				database composed of pairs of gesture motion image and text
3.	Xiaolu Yang, Xuanjing Shen Jianwu Long, Haipeng Chen (College of Computer Science and Technology in Uhrersity, Changchun, un 130012, China Key Laboratory of Symbolic Computanon and Knowledge Engineering of Minuiry of Education, Jilin University, Changchun 130012)	2012	AASRI	An improved median-based Otsu image thresholding Algorithm	The Otsu schema widely used image thresholding technique provides approving results for segmenting a gray level image with only one modal distribution in gray level histogram However, it provides poor results if the histogram of a pay level is non bimodal For enhancing the performance of the Otsu algorithm further in this work an inproved median based Otsu image thresholding algorithm is presented Finally extensive tests are performed and the experiments show that our method obtain more satisfactory results than the original Otsu thresholding algorithm
4.	Mi-Young Cho, Young-Sook Jeong ETRI, 218 Gajeongno, Yuseong-gu, Daejeon, 34129, Korea mycho@etri.re.kr, ysjeong@etri.re.kr	2017	ICACT	Human Gesture Recognition Performance Evaluation for Service Robots	In this paper, they seek to help consumers, robot manufacturers, and gesture recognition engine developers provide comparable results for the gesture recognition capabilities of service robots. This paper deals with the performance evaluation of human gesture recognition technology that is widely used for natural interaction between a human and a service robot. Their test results indicate that

					their database can help consumers, robot manufacturers, and gesture recognition engine developers provide comparable results for the gesture recognition capabilities of service robots.
5.	Soeb Hussain and Rupal Saxena Department of Chemistry Indian Institute of Technology, Guwahati, India soeb.hussain@iitg.ernet.in and upal.saxena@iitg.ernet.in Xie Han, Jameel Ahmed Khan, Prof. Hyunchul Shin Dept. of Electronics and Communication Engineering Hanyang University Sangnok-gu, Korea xiehan@hanyang.ac.kr and shin@hanyang.ac.kr	2017	ISOCC	Hand Gesture Recognition Using Deep Learning	This paper proposes a technique which commands computer using six static and eight dynamic hand gestures. The three main steps are: hand shape recognition, tracing of detected hand (if dynamic), and converting the data into the required command. Experiments show 93.09% accuracy.
6.	Mohd. Baqir Khan, Kavya Mishra,	2017	IEEE	Gesture Recognition using Open-CV	This paper includes introduction to Gesture recognition, its various areas, working of gesture

	Mohammed Abdul Qadeer Department of Computer Engineering, Zakir Hussain College of Engineering & Technology, Aligarh Muslim University, Aligarh 202002, India				recognition system, applications, the proposed idea with its hardware & software implementation tools, the various technologies applied on the idea, benefits of idea, its shortcomings and future scope of gesture recognition system.
7.	Daeha Lee1*, Hosub Yoon2, Jaehong Kim3 1,2Intelligent Cognitive Technology Research Department, ETRI Daejeon, Korea (bigsum, yoonhs, jhkim504)@etri.re.kr	2016	ICROS	Continuous gesture recognition by using gesture spotting	In this paper, to extract a meaningful gesture portion in an online situation, we introduce a method that can distinguish the start and end of a gesture. Then, we describe the method of recognizing an extracted gesture. Using the “Stand” posture, they detect the start and end points of a meaningful target gesture, and they perform the recognition process using front/right/left MHIs; thus, the gesture recognition result is improved. In the future, we will apply the proposed method to various fields.
8.	Dr. Suchart Yammen Senior IEEE Member (sucharty@nu.ac.th) Dr. Sureerat Tang	2019	IEEE	IoT based speed control of Smart Fan	This paper clarifies an efficient speed control strategy for an electric fan module utilizing SmartAndroid or IOS phone. Espino and Arduino's circuits are utilized in this procedure to accomplish the target of this paper. Wi-Fi connection is utilized as a communication protocol between fan, Espino board, and advanced mobile phone. The objective is to form a current fan

	(sureerat@tjsupply.co.th) Mahesh Kumar Reddy Vennapusa (maheshkumarr59@email.nu.ac.th)				into a brilliant fan; along these lines, the client can direct the fan speed from his or her stride. The final prototype fan model significantly improves convenience for senior residents and crippled people. The proposed strategy is straightforward and gives space to assist improvement.
9.	Ola Younis, Waleed Al-uaimy, Fiona Rowe and Mohammad H. Alomari	2018	IEEE	Real-time Detection of Wearable Camera Motion Using Optical Flow	The proposed approach has been tested on a real first-person perspective video captured by a wearable camera. The experimental results demonstrate that the proposed method classifies the type of motion successfully in real-time and can be used as part of low-cost wearable solutions for various forms of vision loss assistive technologies. Promising performance results of 84% correct states for camera motion detection were obtained.
10.	Sunila Gollapudi S. Gollapudi, Learn Computer Vision Using OpenCV, https://doi.org/10.1007/978-1-4842-4261-2_2	2019	S. Gollapudi	OpenCV with Python	This is a lesson manual about OpenCV using python. The manual will cover setting up your system with OpenCV and the Python libraries, understanding key modules and out-of-box functions for computer vision implementations, and learning the syntax for scaling up.
11.	Jan Flusser, Senior Member, IEEE, Sajad Farokhi, Cyril Höschl IV, Tom´aˇs Suk, Barbara Zitov´a, and Matteo Pedone	2015	IEEE	Recognition of Images Degraded by Gaussian Blur	This paper proposes a new theory of invariants to Gaussian blur. It proves that the moments of the primordial image are invariant to Gaussian blur and it derives recursive formulas for their direct computation without actually constructing the primordial image itself. The proposed method is significantly faster and more robust

					to additive noise while its recognition rate in noise-free cases is fully comparable to the Zhang's distance. Their method handles also an anisotropic Gaussian blur and is even able to compare images of different sizes.
12.	Tae Hyun Kim, Seungjun Nah, and Kyoung Mu Lee	2017	IEEE	Dynamic Video Deblurring using a Locally Adaptive Linear Blur Model	The paper introduces a novel method that removes general blurs in dynamic scenes which conventional methods fail to. It has inferred bidirectional optical flows to parametrize motion blur kernels and estimated the scales of Gaussian blurs to approximate defocus blur kernels. Therefore the proposed method could handle general blurs, by estimating a pixel-wise different blur kernel. In addition, it proposes a new single energy model that estimates optical flows, defocus blur maps and latent frames, jointly.

Technology Used:

1. Segmentation of Fingers and extraction of outline of finger.
2. Motion detection using absdiff() function from CV2 library.
3. Otu's Thresholding to convert colorful image into binary image.
4. Gaussian Blur.

Final Outcome:

Our proposed system will have the outcome of product-based. The user will be able to work on the technology affiliated with the concept of smart fans. They will trigger the events from gesture recognition, which will give the response. Their Motion will be detected from the camera and conclude if there is anyone present in the room.

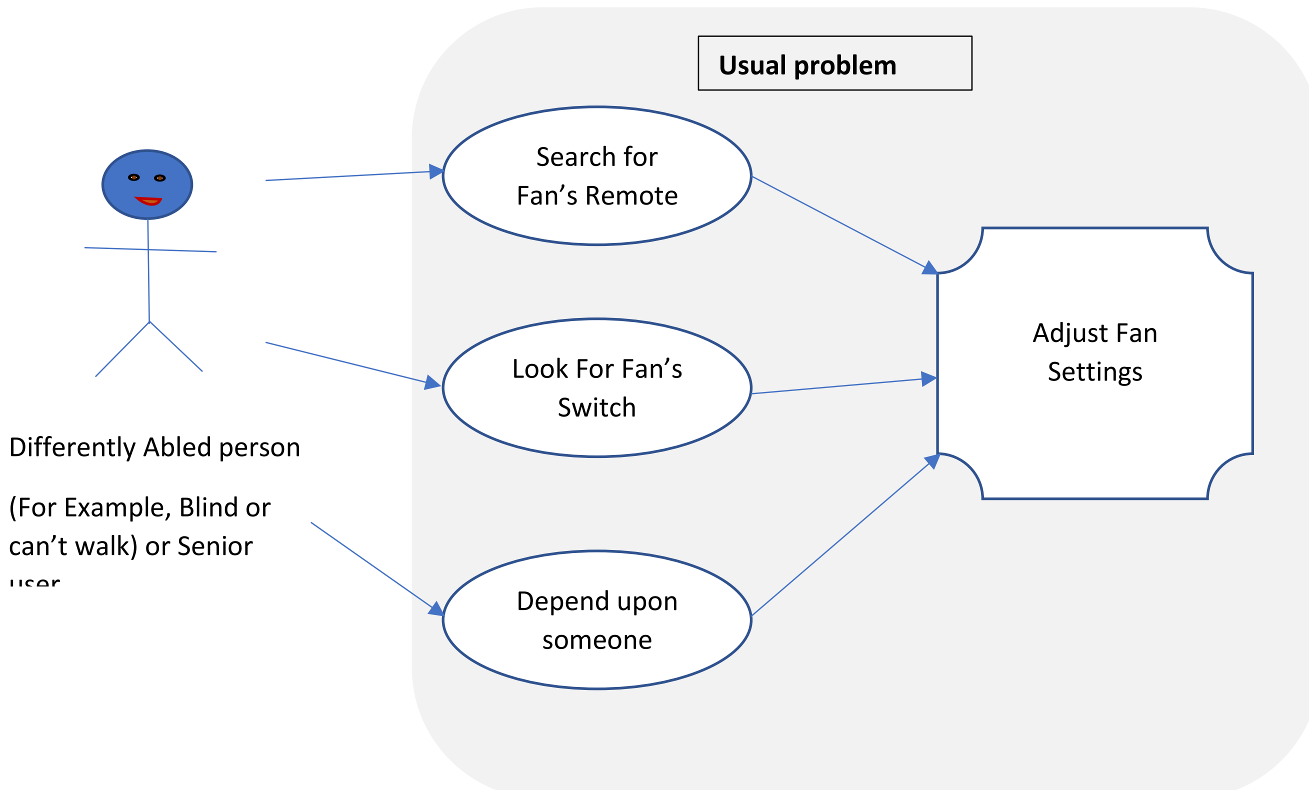
Working

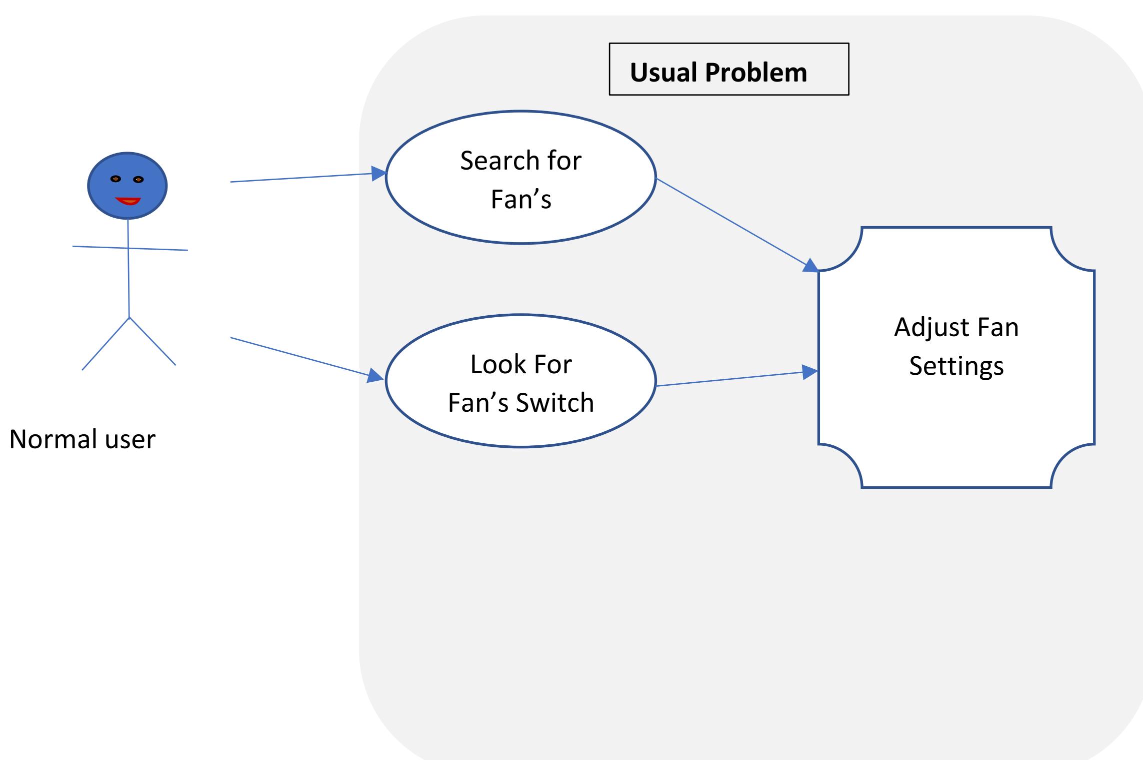
With growth of technology in various fields, there has been a lot of digital improvements in various sectors. Now we are exploring various options to combine technology to improvise the equipment's. Our proposed system is product based focusing on equity. Differently-abled people or Senior Citizens in using the equipment's faces problems. But with our Proposed model we can overcome this problem.

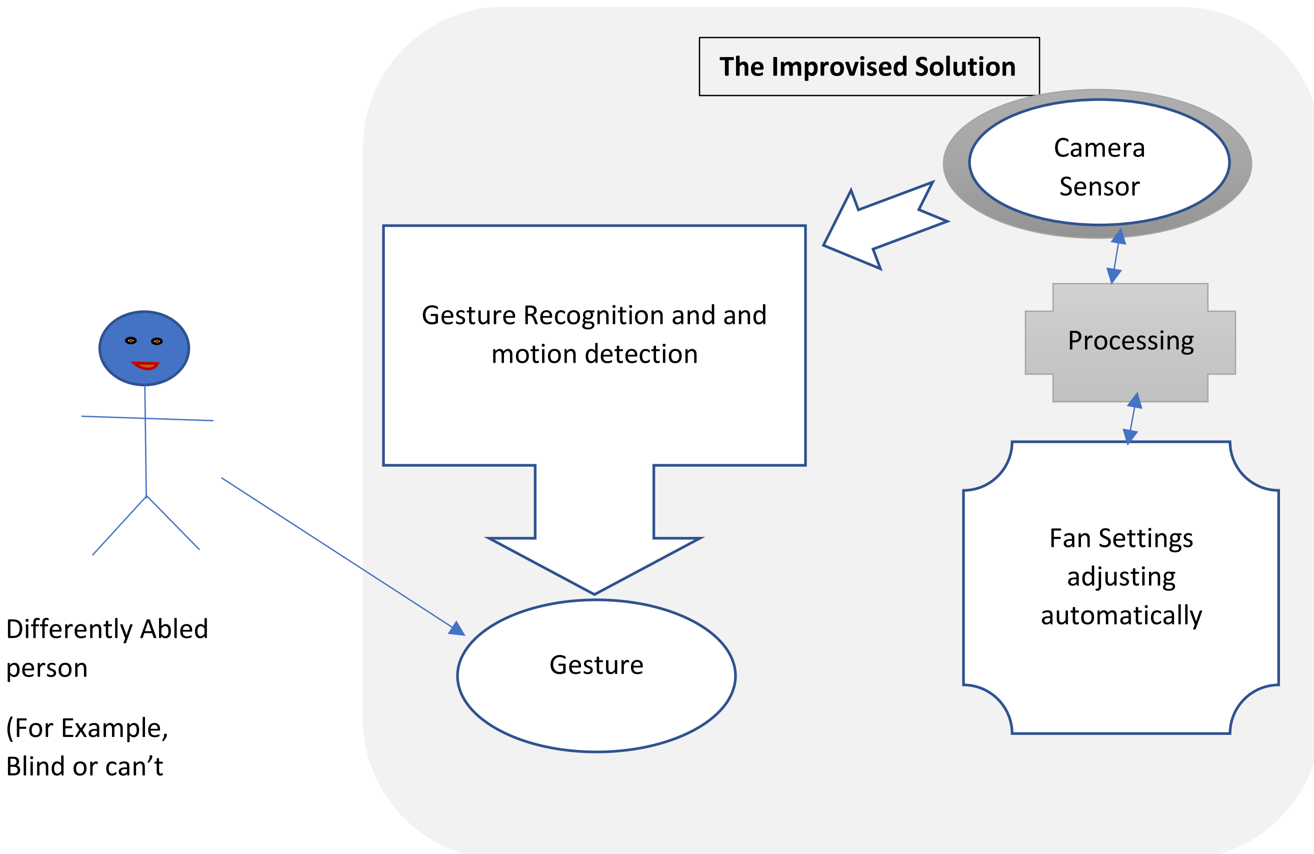
Smart fans and ordinary fans: Normally with ordinary fans, user have to look for remote or manually rotate the switch for operating it. The technology of switch is not mobile. However, remotes are mobile. But Remotes are liabilities, which needs to be taken care of. Our proposed model is about using the gesture pointed at camera. The camera could be attached to fan if the fans are wall fans, but for ceiling fans CCTV cameras can be used. With motion detection technology used, the Smart fans technology could be further expanded into secret surveillance system.

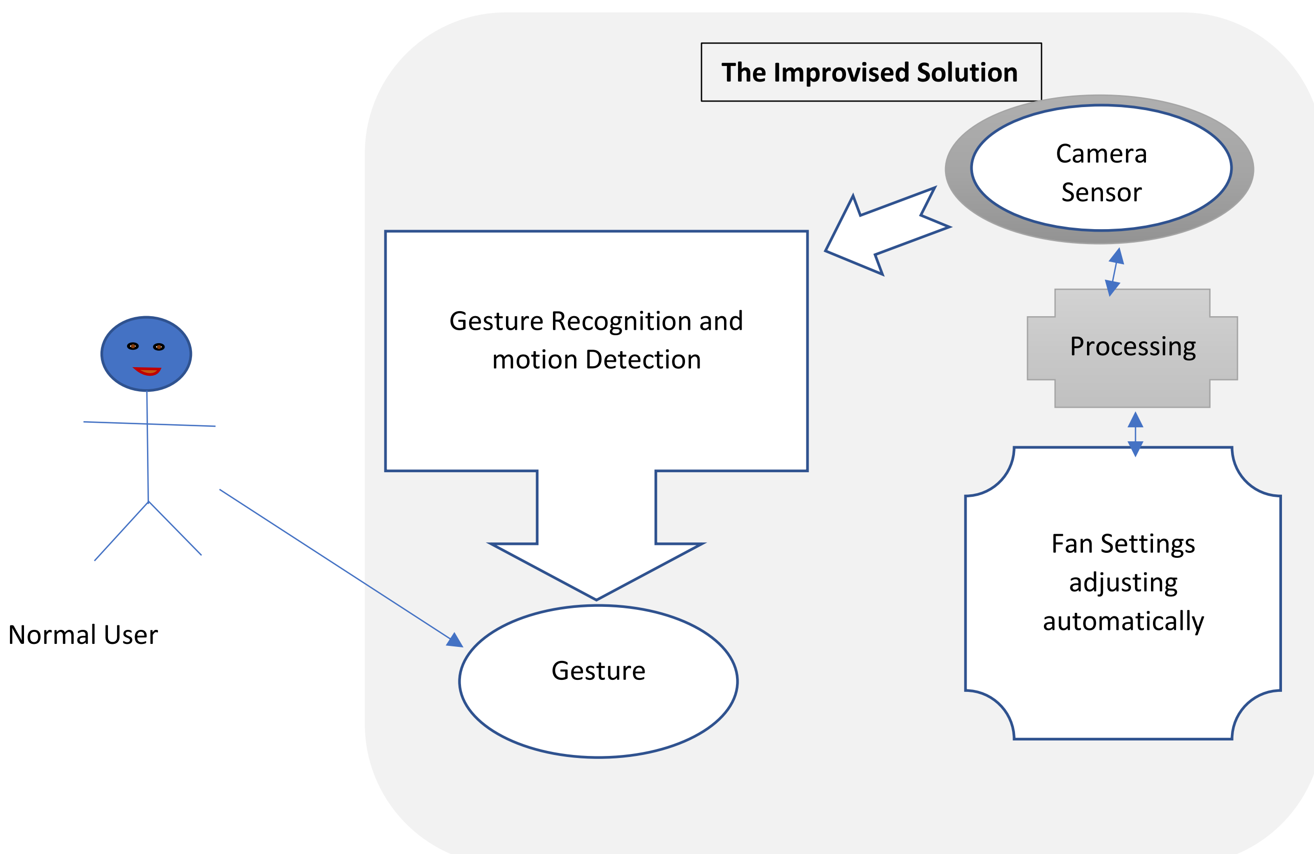
- Gesture Recognition: It is staple method of interaction especially for the deaf and the blind people.
  - Segmentation: The image captured by the camera is divided into multiple segment.
  - Otsu thresholding: It helps to return a single intensity threshold that separate pixels into two classes, foreground and background.
  - High-boost filter: It is used for amplifying high frequency components of signals and image which results in image sharpening.
  - Cosine angle:
  - Contours counting: contours represent the shape of the object found in a image that is captured by camera.
- Motion Detection: It detects the motion of the user and read it to give the corresponding response what the user wants.
  - Capture Frame: frame capture using OpenCV
  - Gaussian Blur: As sometimes the capture motions of the user can be blurred deu to the incorrect focus of the camera so to reduce the image noise and reduce detail Gaussian blur is used so that the motions is clearly visible.
  - Finding contours: we used OpenCV tool to extract contours and find its movement.

Use Case Diagram

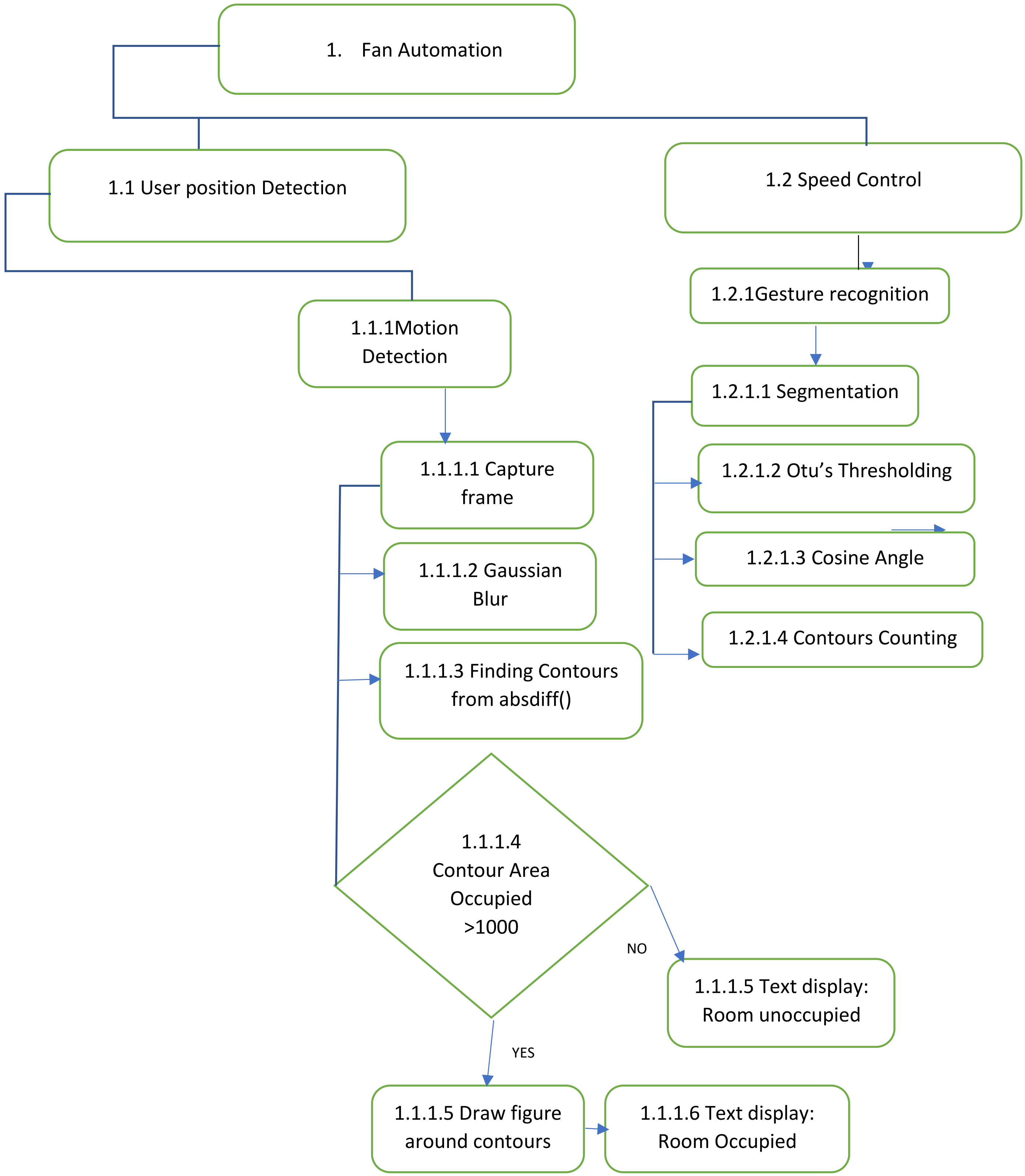




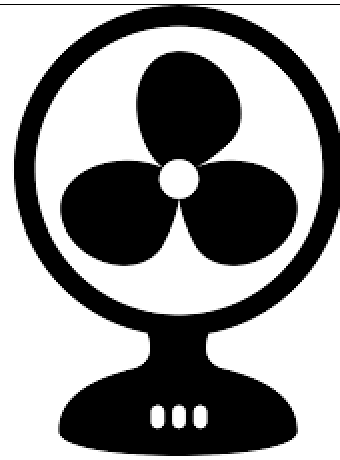
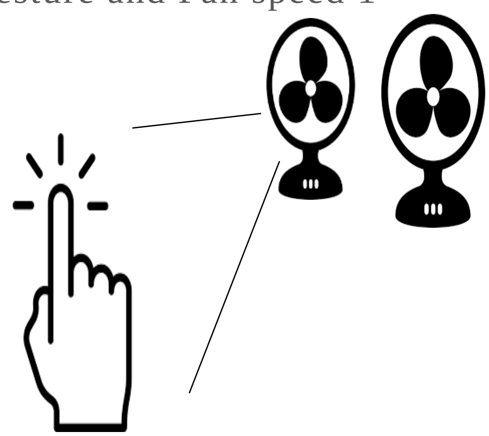

The Improvised Solution
Camera Sensor
Gesture Recognition and and motion detection
Processing
Fan Settings adjusting automatically
Differently Abled person
(For Example, Blind or can't
Gesture

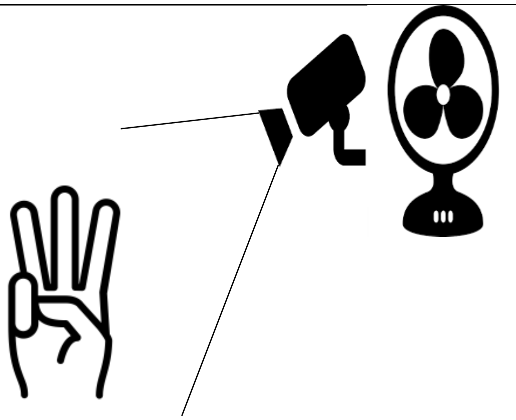
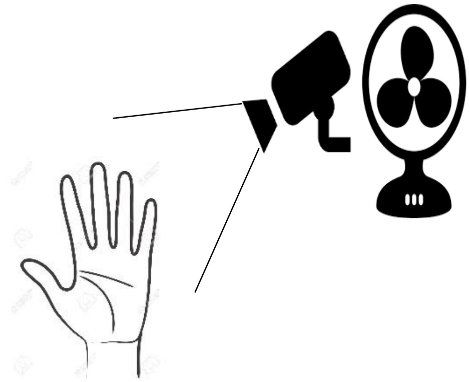

The Improvised Solution
Camera Sensor
Gesture Recognition and motion Detection
Processing
Fan Settings adjusting automatically
Normal User
Gesture

Hierarchical Task Analysis



Story Boarding

 User	 Smart Fan
Camera Sensor Detecting User motion 	Camera sensor recognizing the gesture and Fan speed 1 

 Camera sensor recognizes 3 fingers and fan speed 3	 Camera sensor recognizes 5 fingers and closes.

Primary Stakeholder Profiles

Context of Use

Motivation	**Social Environment**	**Technical Environment**
☐Discretionary	☐Public	☐Networked
☒Mandatory	☒Personal	☒Isolated
Frequency of Use	☐Collaborative	☐Wired
☐Non-User	☐Individual	☒Wireless
☐Infrequent		
☒Frequent	☒Work	☒Intranet
	☒Entertainment	☐Extranet
USER		☐Internet
☒Beginner	☐Synchronous	
☒Intermediate	☒Asynchronous	☒PAN
☐Expert		☐LAN
	Physical Environment	☐MAN
Task Nature	☒Indoor	☐WAN
☒Mission Critical	☐Outdoor	
☐Calm		☒Fixed
Interaction Mode	Auditory (Noise Level)	☐Mobile
☒Direct	***1*** 2 3 4 5	☒Peripherals
☐Indirect		☐Contained
	Visual Quality	
☒Continuous	1 2 ***3*** 4 5	**Haptic**

☐Intermittent

☐Constrained

☒Free

Cognitive Ability

Educational level

☐Elementary

☒Middle School

☒Graduate School

Computer Literacy

system

1 ***2*** 3 4 5

Application

1 ***2*** 3 4 5

Domain Knowledge

1 ***2*** 3 4 5

Cognitive Style

☒Visual

☐Auditory

☒Graphical

☒Linguistic

Typing Skill (Words per Minute)

Novice _______________

Intermediate___________

Expert_______________

Physical Ability

Visual Blind

☒20/200

☒20/100

☒20/70

☒20/50

☒20/40

☒20/30

☒20/25

☒20/20

Color Vision

☒Trichromatic

☒Protanomaly

☒Deuteranomaly

Auditory

1 2 3 4 5

Haptic

☒Disabled: *Blind, Deaf, Dumb, Old age*

☒Fully Functional

Individual Profile

Age

☐Early Childhood

☐Childhood

☐Preteen

☐Teen

☐Young Adult

☐Adult

☐Middle Age

☐Senior

Gender

☐Male

☐Female

Occupation

Interests

Country:

Region:

Language:

Ethnicity:

Religion:

Socio-Economic:

Tasks identification and their categorization

Easy	Moderate	Tough
Gesture Recognition		
Otu’s Thresholding	High-Boost Filter	Segmentation
	Cosine Angle	Contours Counting
Motion Detection		
Capture frame	Circle around contours	Finding Contours from dilated frame
Gaussian Blur	Rectangle around contours	Contour Area
		absdiff()

REVIEW 2

Two major tasks involved:

- a) **Hand gesture recognition**
- b) **Motion Detection**

a) Hand Gesture Recognition

1. <u>Gesture outside the Visual Area</u>
2. <u>Gesture inside the visual Area</u>

b) Motion Detection

1. <u>Movement outside the visual area</u>
2. <u>Movement inside the visual area</u>

a) Hand Gesture Recognition

1. Gesture Outside the Visual Area

   Model Human Processor

   Total Time= Tp+Tc+Tm

   =100+(5)70+70

   =520ms

   KLM

SN	Description	Symbol	Time (ms)
1	Mentally preparing for the gesture	M	1.35
2	Pointing the gesture	P	1.10
3	Moving hand to visual area	H	0.4
4	Verifying the Recognition	M	1.35
			4.2

GOMS analysis –

– Method for goal: Mentally preparing for the gesture
- Step 1. Decide the gesture.
- Step 2. Return with goal accomplished.

– Method for goal: Pointing the gesture
- Step 1. Point the gesture to the visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Moving hand to visual area
- Step 1. Move your gestured hand to visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Verifying the Recognition
- Step 1. Verify the Recognition.
- Step 2. Return with goal accomplished.

2. Gesture Inside the visual Area

   Model Human Processor

   Total Time= Tp+Tc+Tm

   =100+(5+1)70+70

   =590ms

   KLM

SN	Description	Symbol	Time (ms)
1	Mental	M	1.35
2	Moving hand to visual area	H	0.4
3	Mentally preparing for the gesture	M	1.35

4	Pointing the gesture	P	1.10
5	Verifying the Recognition	M	1.35
			5.55

GOMS analysis –

– Method for goal: Moving hand to visual area
- Step 1. . Point the gesture to the visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Mentally preparing for the gesture
- Step 1. Decide the gesture.
- Step 2. Return with goal accomplished.

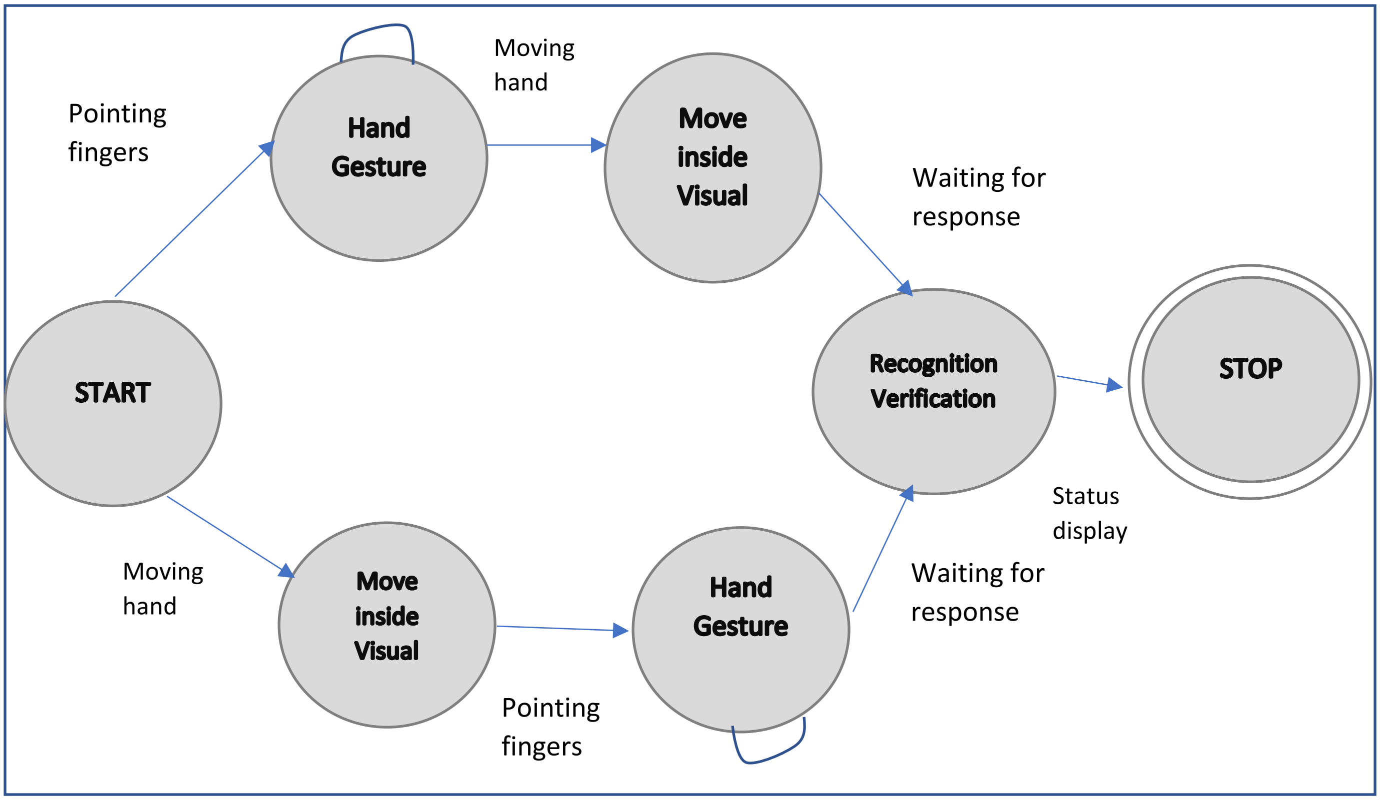
– Method for goal: Pointing the gesture
- Step 1. Point the gesture to the visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Verifying the Recognition
- Step 1. Verify the Recognition.
- Step 2. Return with goal accomplished.

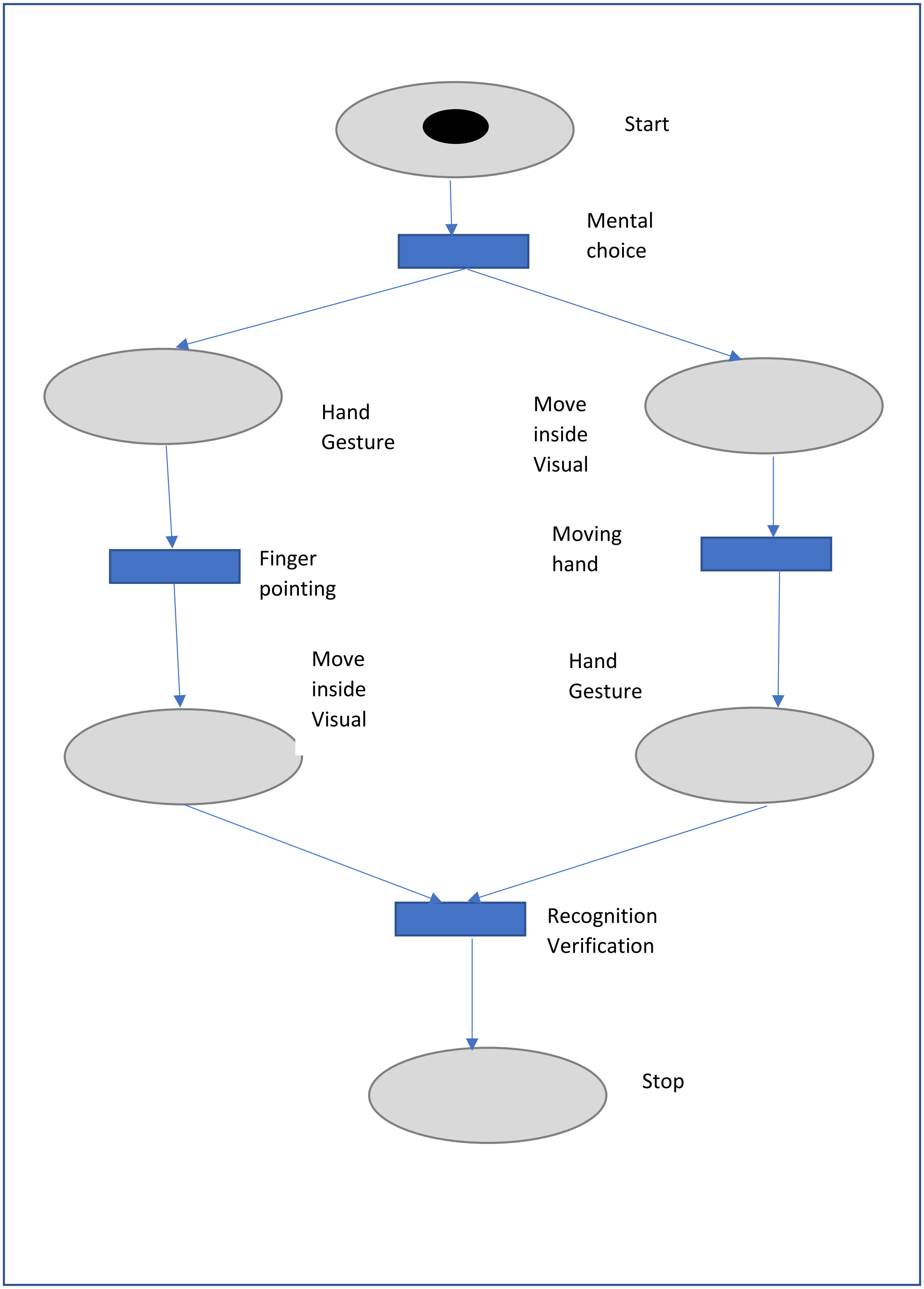
Comparison

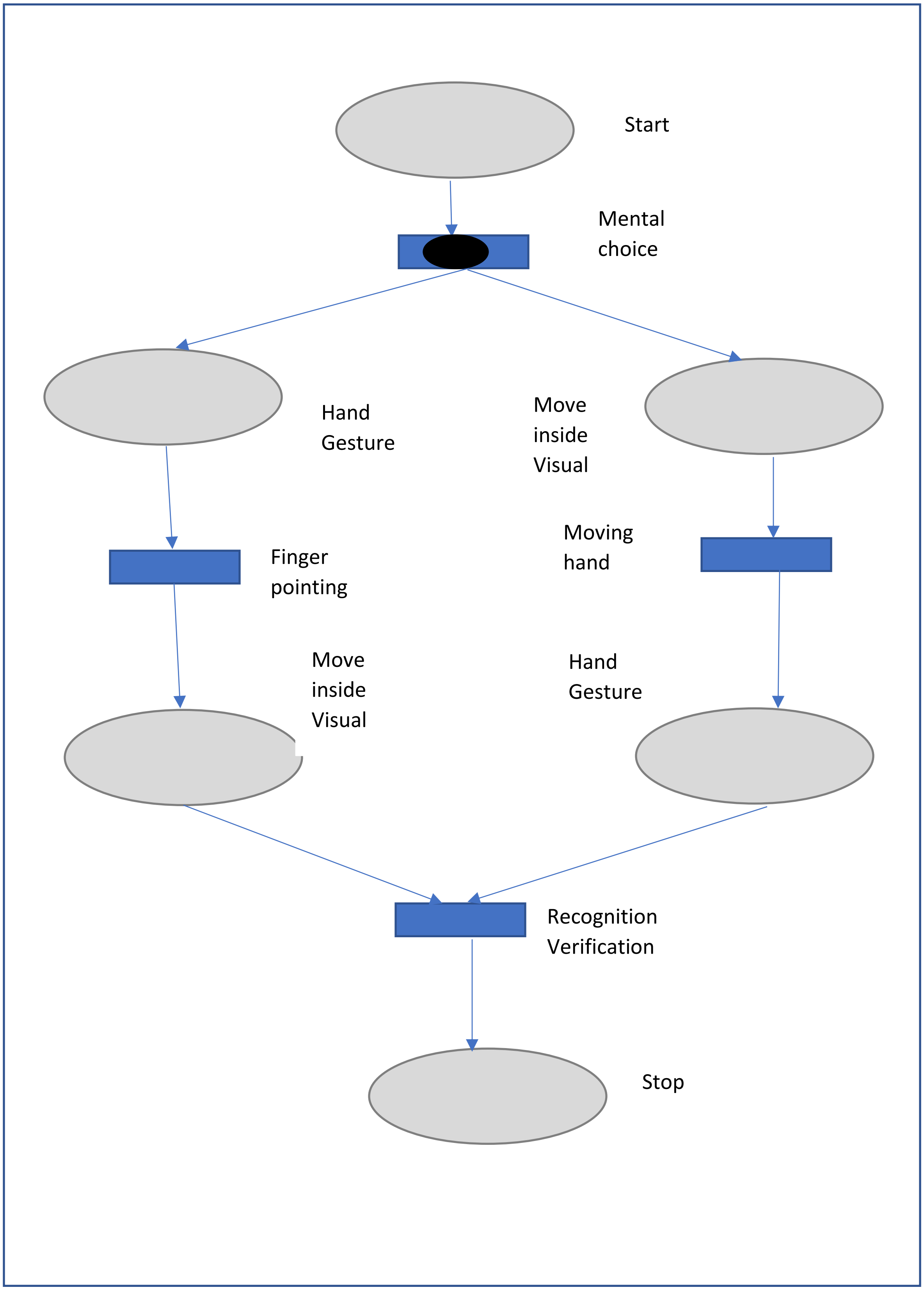
	MHP	KLM
Gesture outside the Visual Area	520ms	4.2
Gesture Inside the visual Area	590ms	5.55

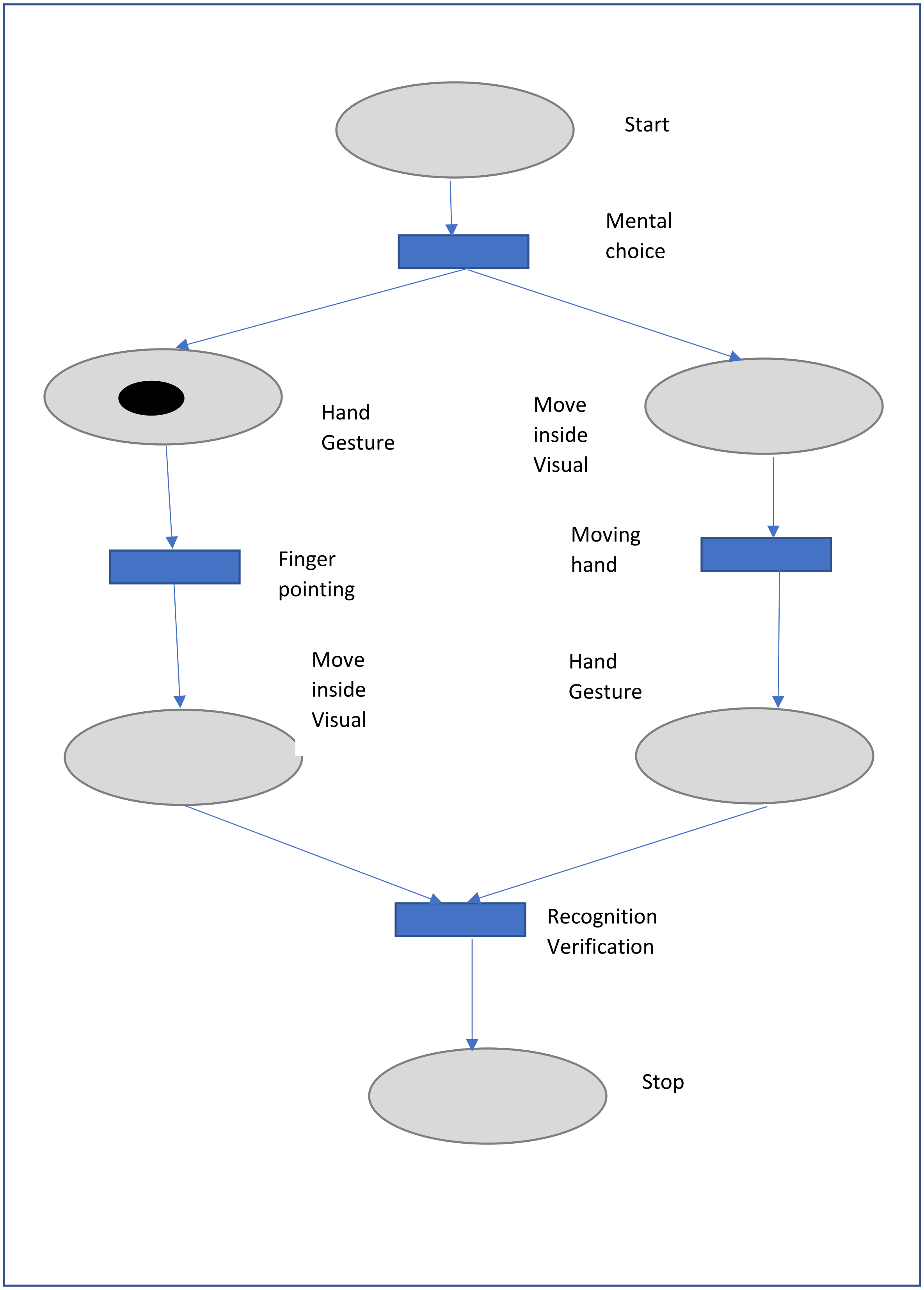
State Transition Network

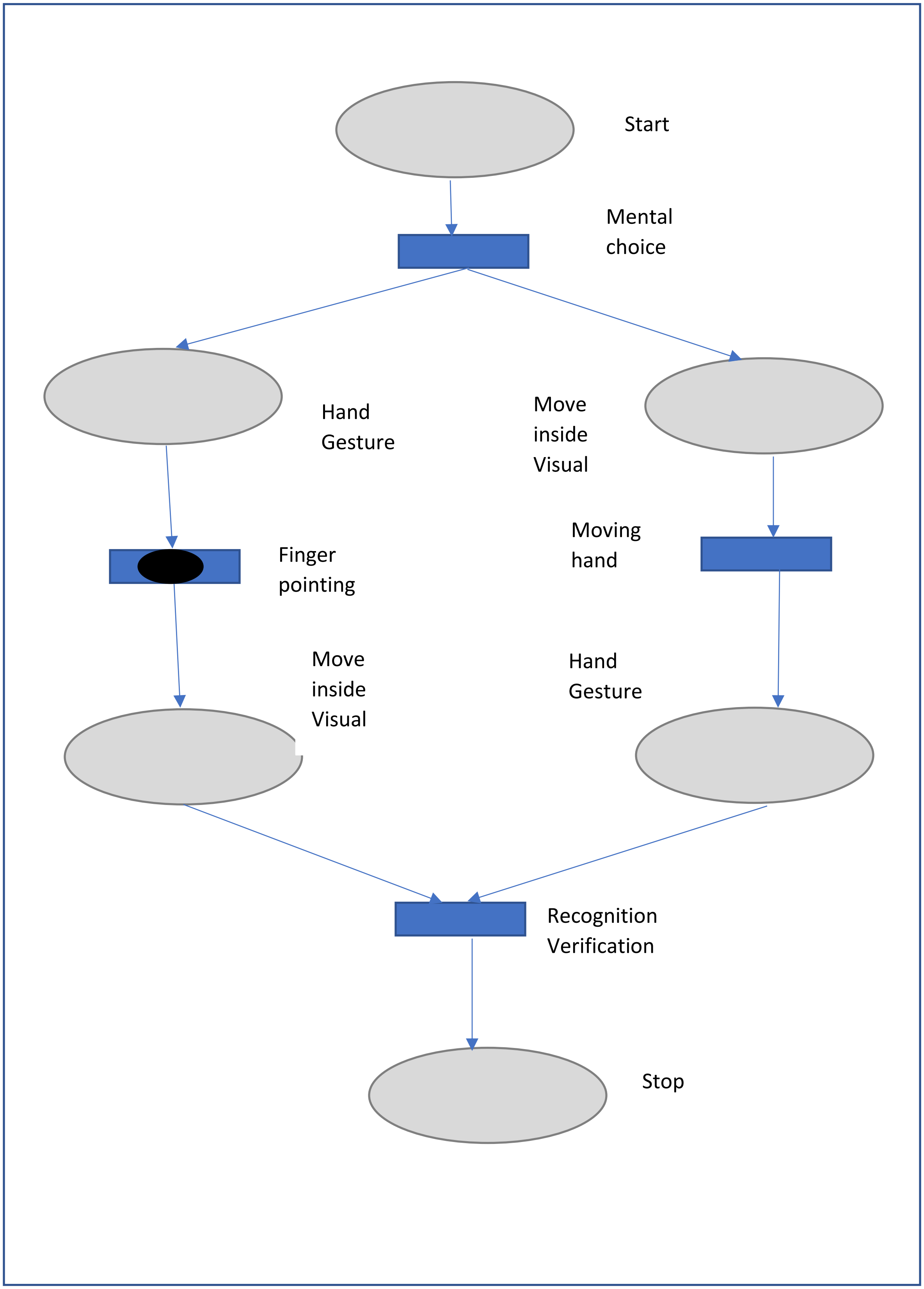


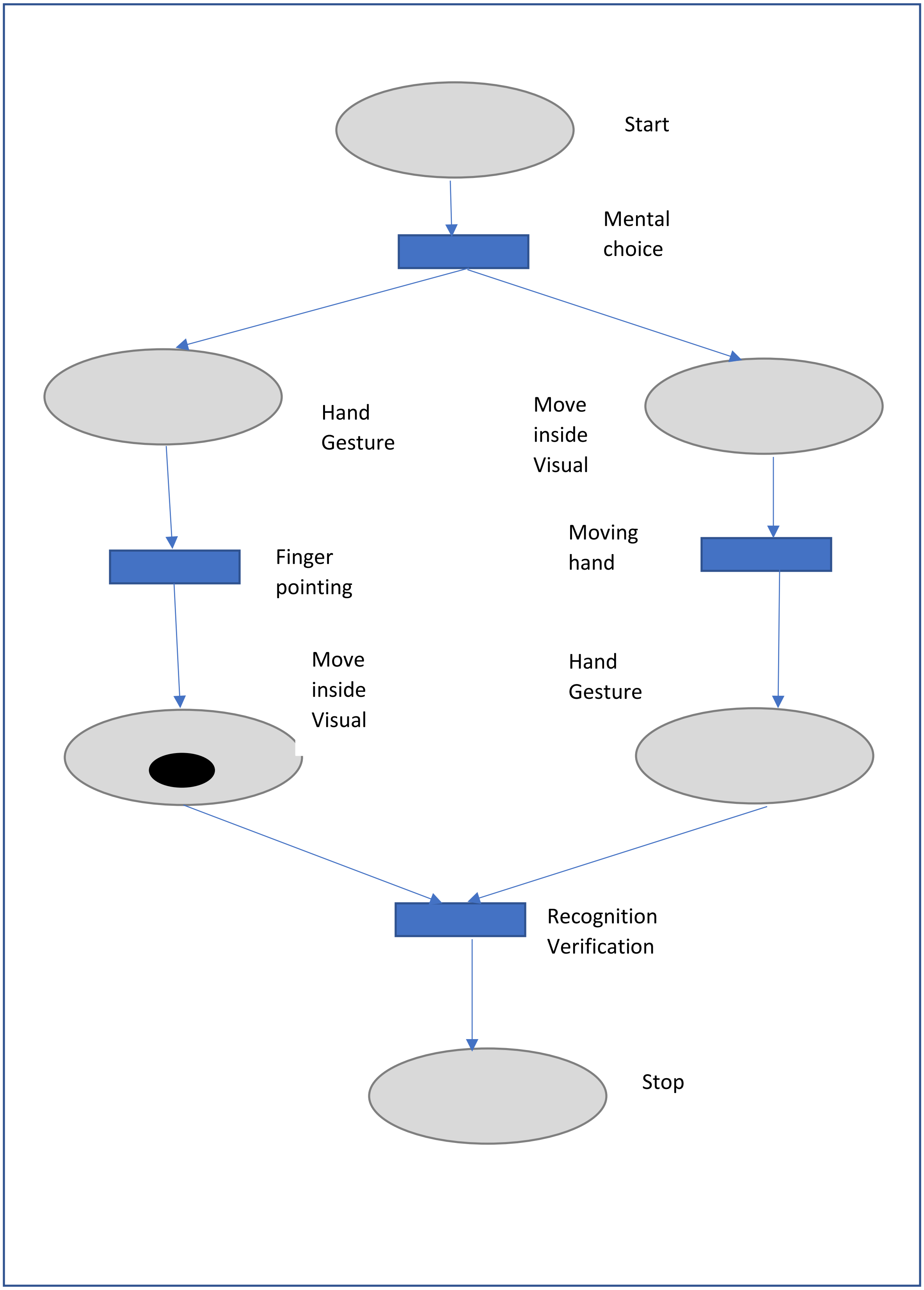
Petri-Mets

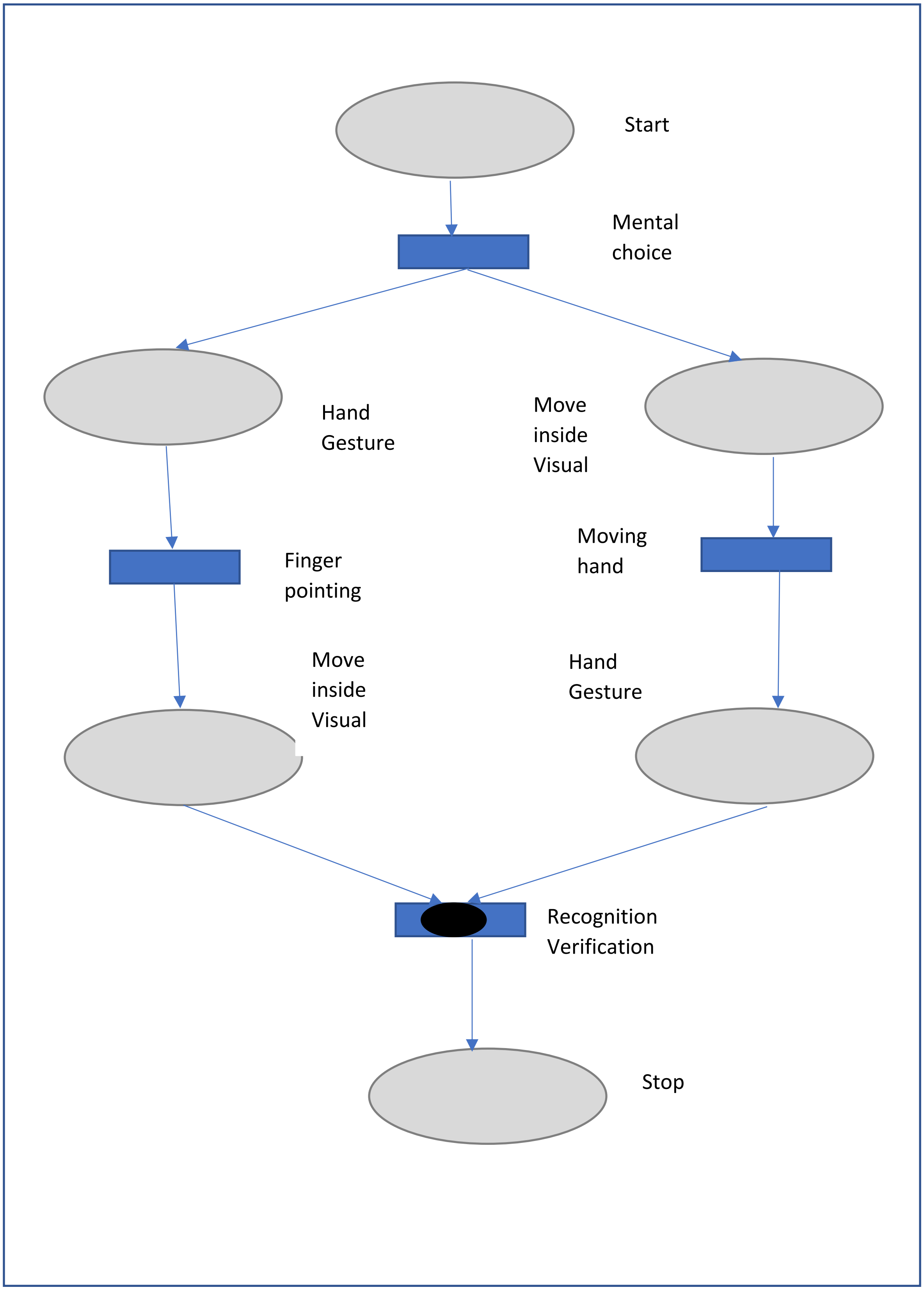

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

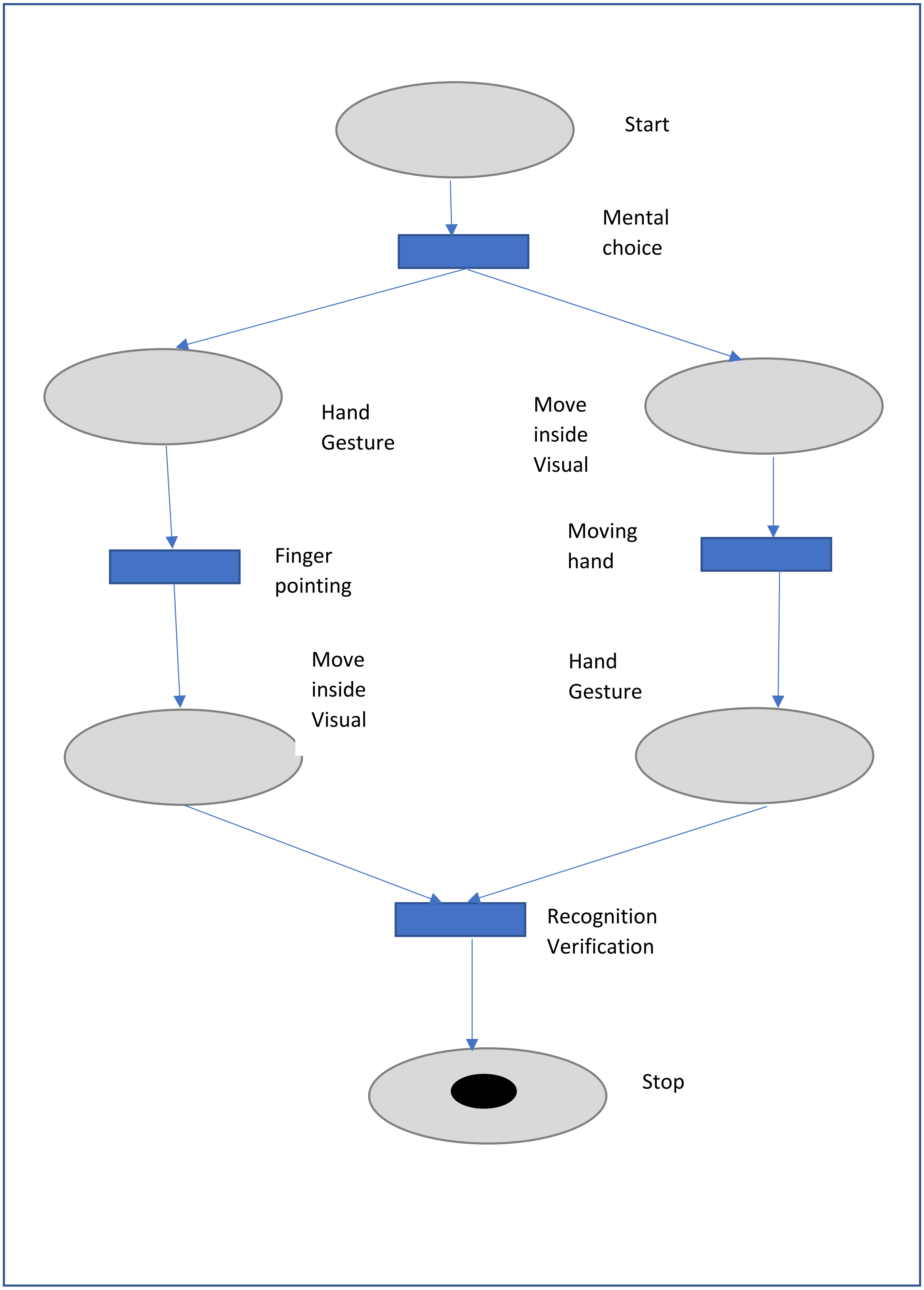

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

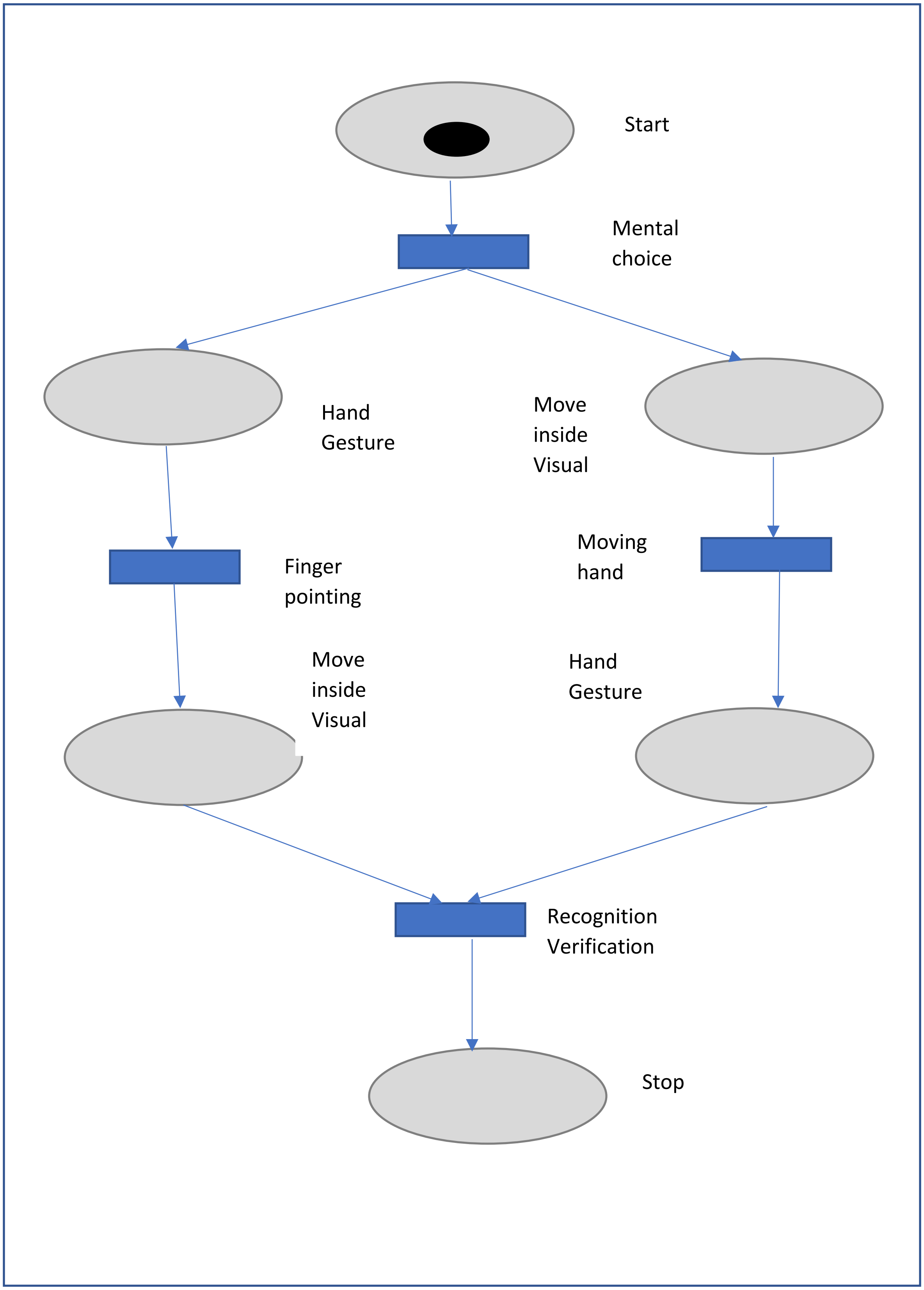

Start
Mental choice
Hand Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Hand Gesture
Recognition Verification
Stop

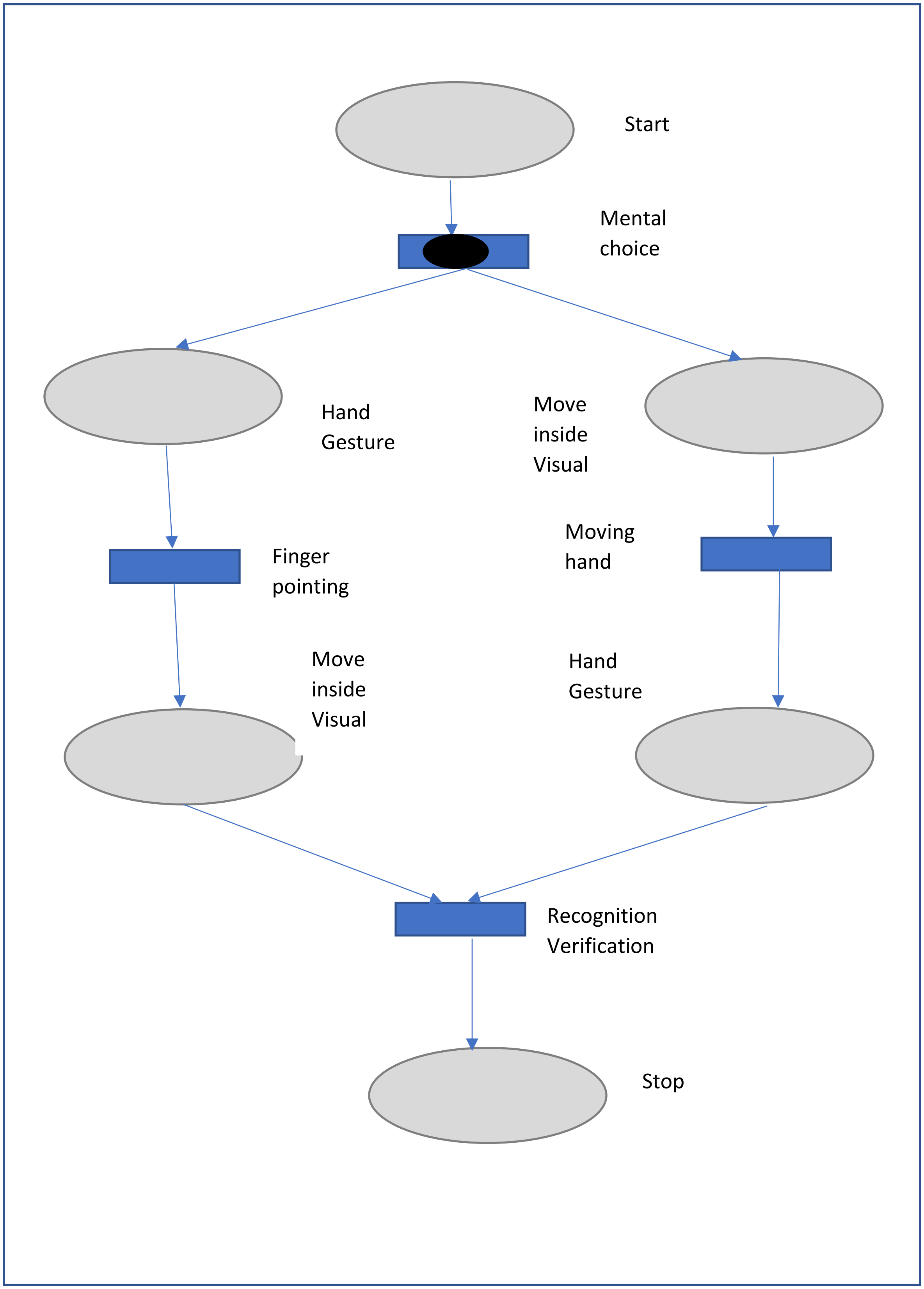

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

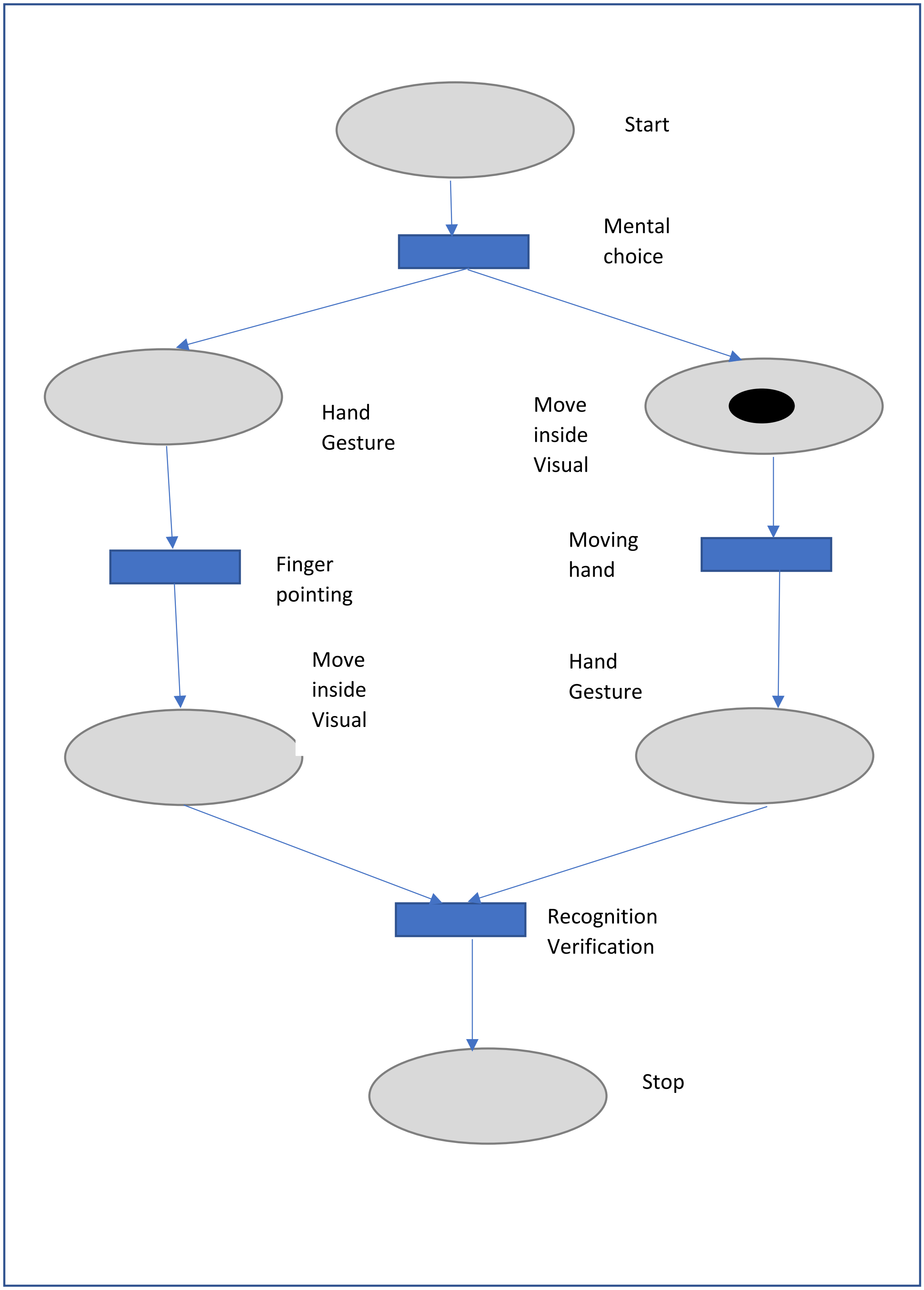

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop


Start
Mental choice
Hand Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Hand Gesture
Recognition Verification
Stop

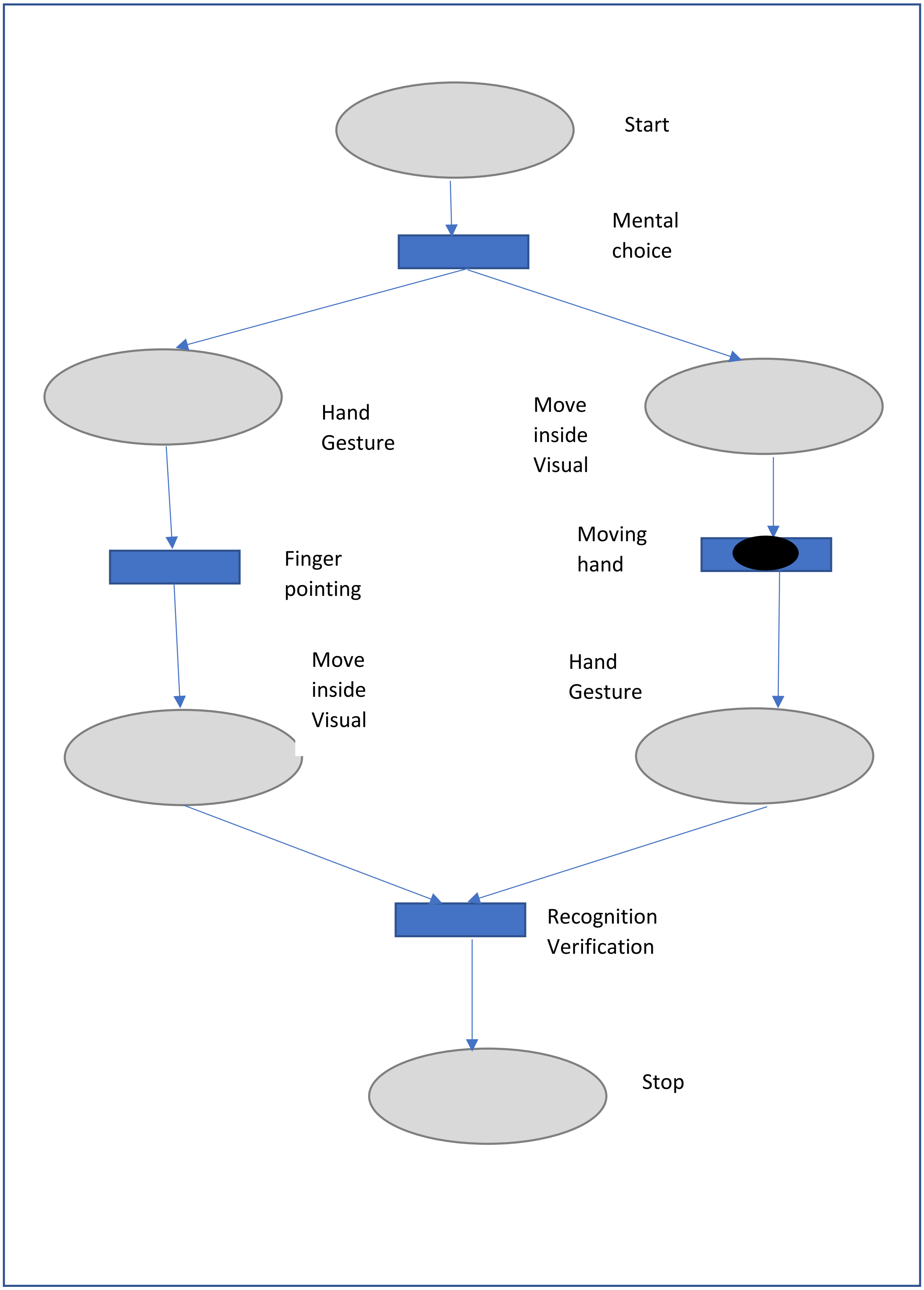

Start
Mental choice
Hand Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Hand Gesture
Recognition Verification
Stop

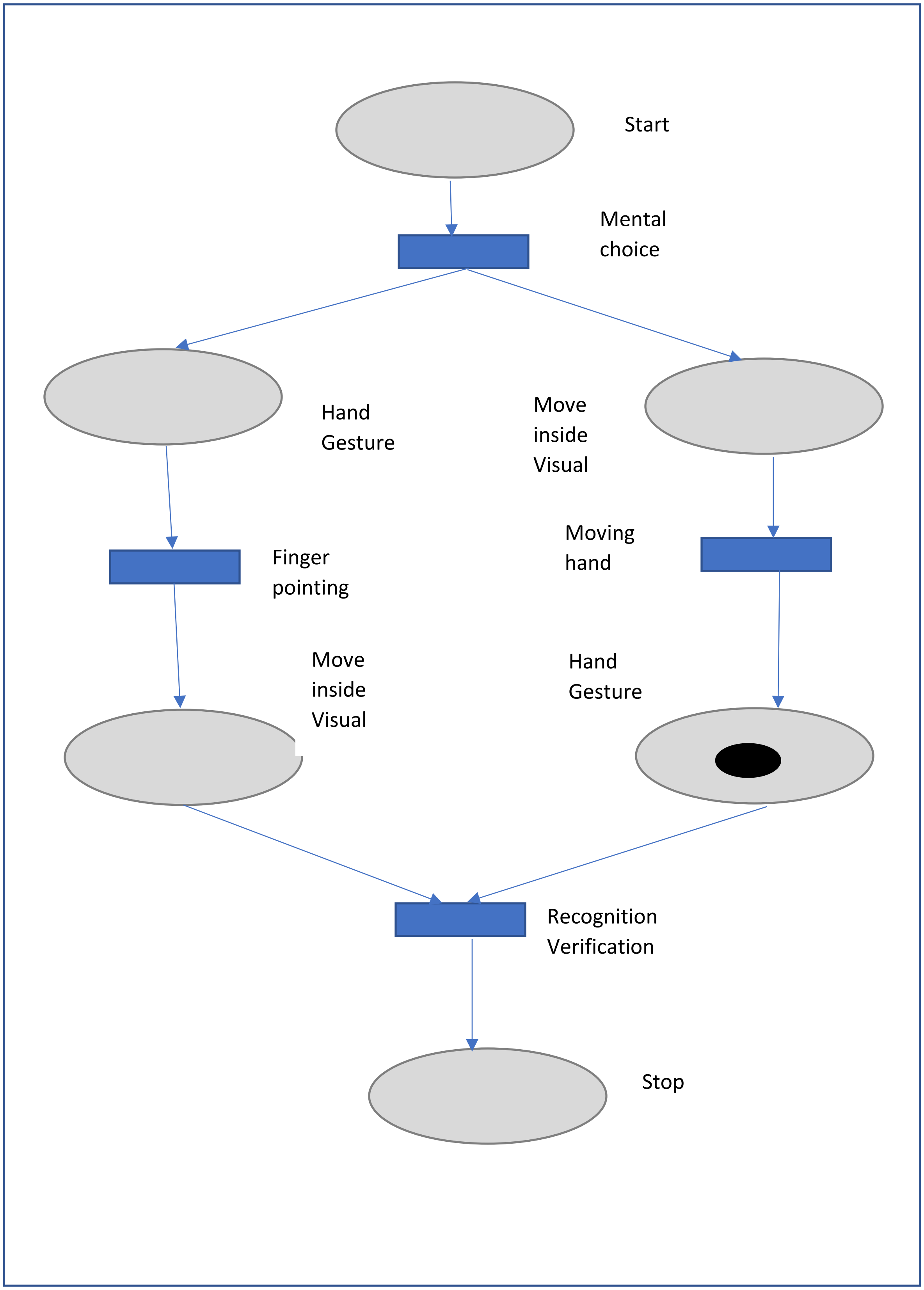

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Moving
hand
Finger
pointing
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

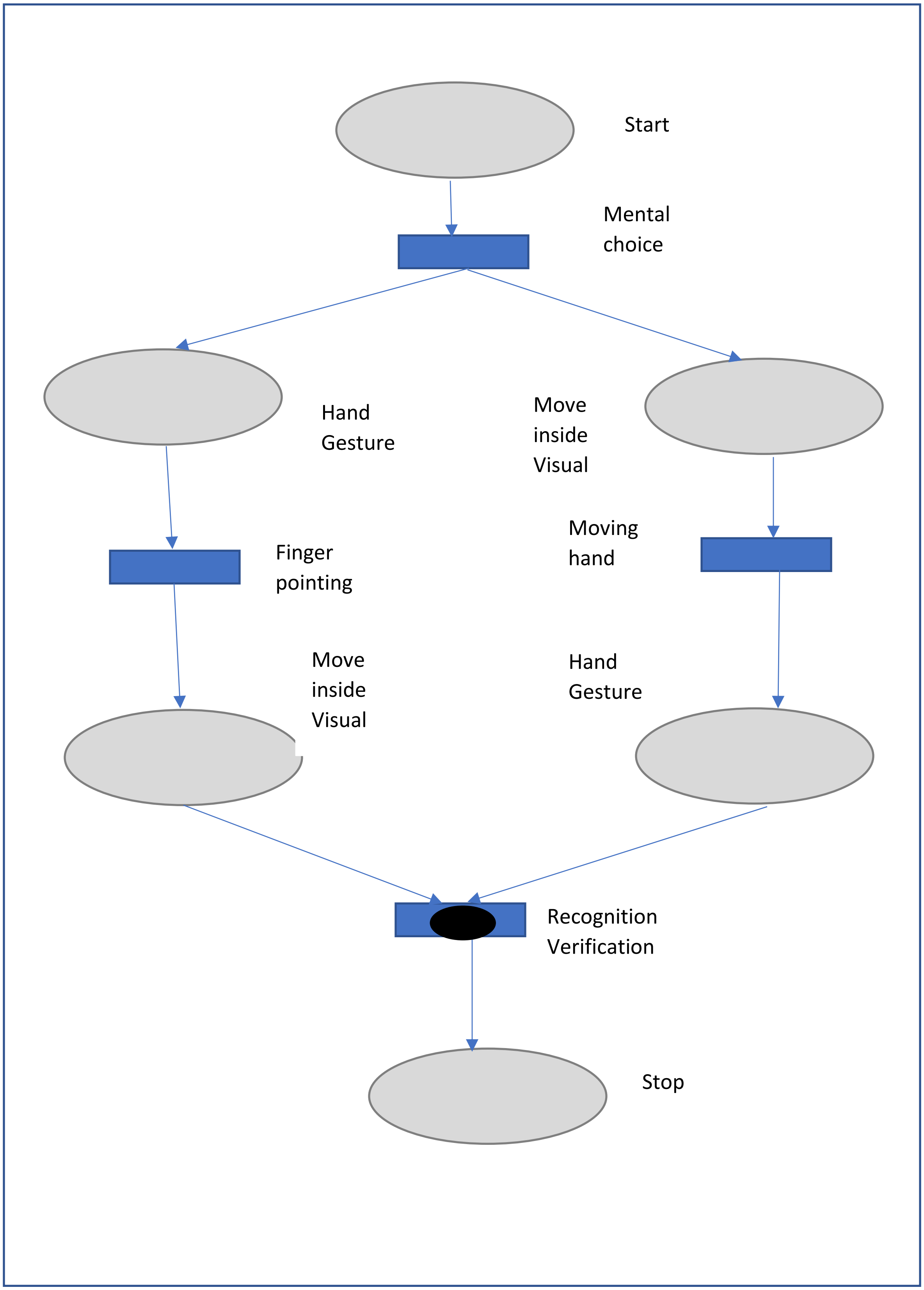

Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

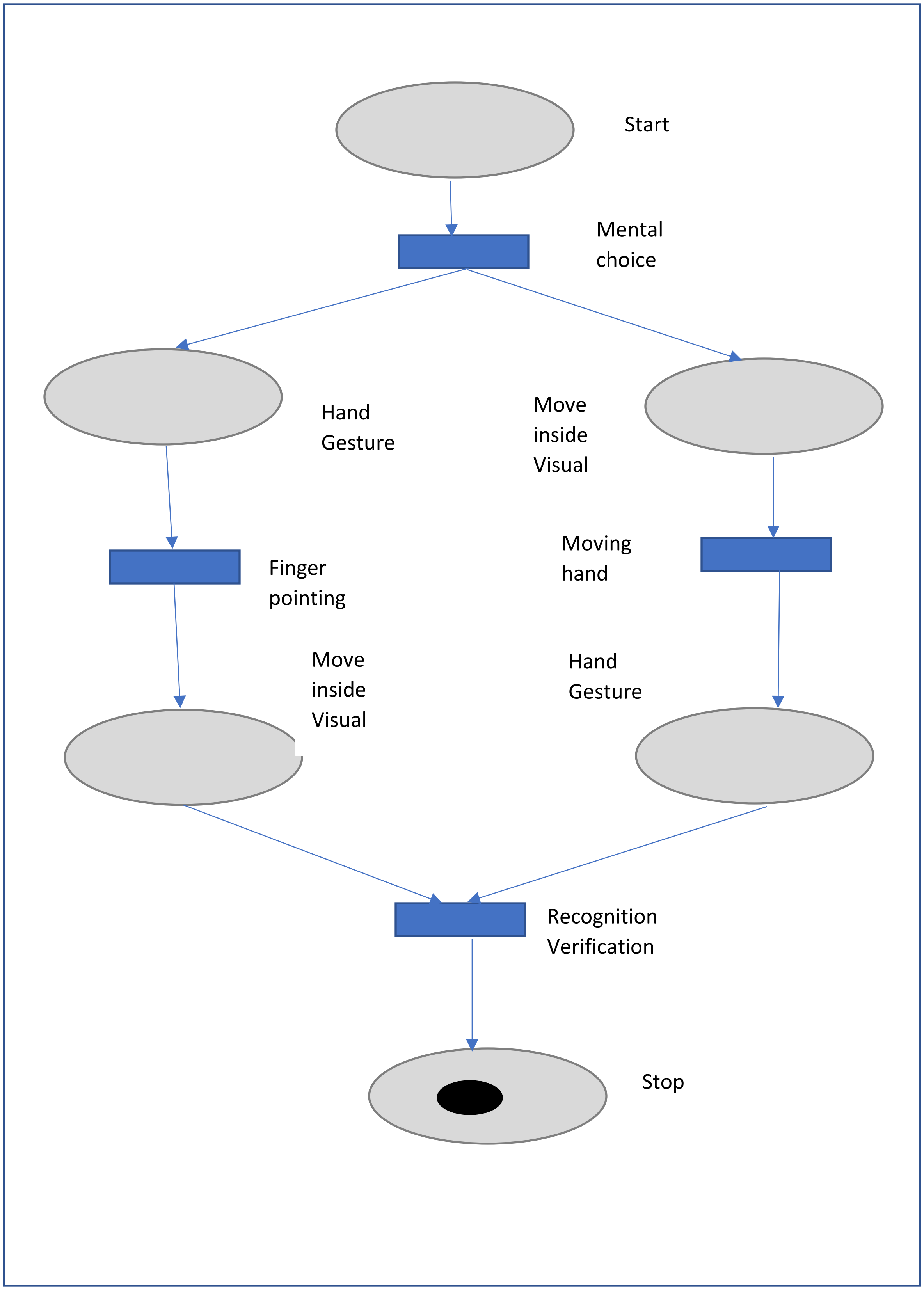


Start
Mental choice
Hand Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Hand Gesture
Recognition Verification
Stop

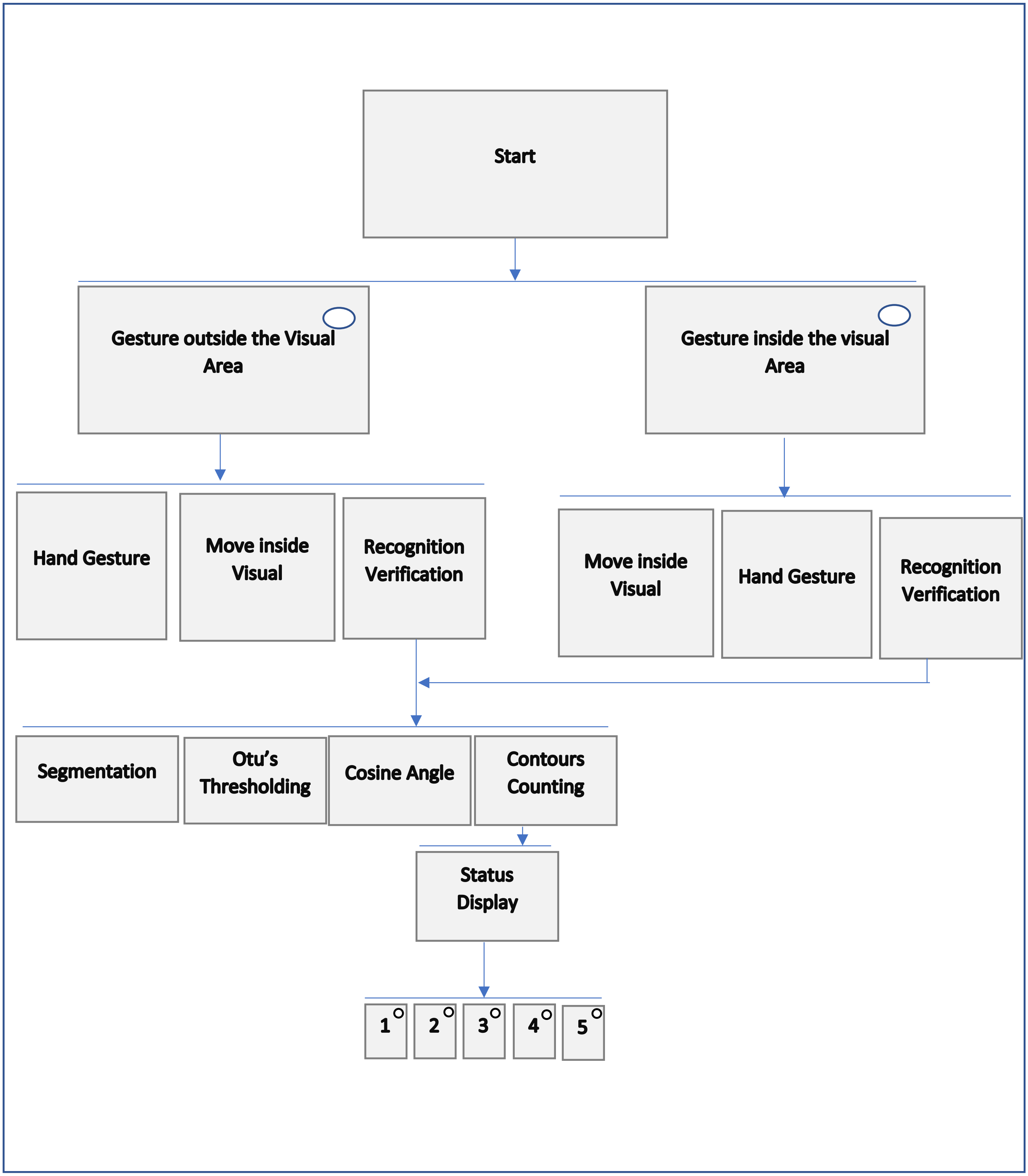

Start
Mental choice
Hand Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Hand Gesture
Recognition Verification
Stop


Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop


Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop


Start
Mental
choice
Hand
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Hand
Gesture
Recognition
Verification
Stop

Jackson Structured diagram



b) Motion Detection

1. Movement inside the visual area

Model Human Processor
Total Time= Tp+Tc+Tm
=100+(5+1)70+70
=590ms

KLM

SN	Description	Symbol	Time (ms)
1	Mental	M	1.35
2	Moving hand to visual area movement	H	0.4
3	Mentally preparing for the gesture	M	1.35
4	Pointing the gesture movement	P	1.10
5	Verifying the Detection	M	1.35
			5.55

GOMS analysis – Movement inside the visual area
– Method for goal: Moving hand to the visual area

- Step 1. Move your gestured hand to visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Pointing gesture movement

- Step 1. Point the gesture to the visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Verifying the motion detection

- Step 1. Verify the Detection
- Step 2. Return with goal accomplished..

2. Movement outside the visual area

Model Human Processor
Total Time= Tp+Tc+Tm
=100+(5)70+70
=520ms

KLM

SN	Description	Symbol	Time (ms)

1	Mentally preparing for the gesture	M	1.35
2	Pointing the gesture movement	P	1.10
3	Moving hand to visual area movement	H	0.4
4	Verifying the detectoin	M	1.35
			4.2

GOMS analysis – Movement outside the visual area

– Method for goal: Mentally preparing for the gesture

- Step 1. Decide the gesture.
- Step 2. Return with goal accomplished.

– Method for goal: Pointing the gesture

- Step 1. Point the gesture to the visual area.
- Step 2. Return with goal accomplished.

– Method for goal: Moving hand to visual area

- Step 1. Move your gestured hand to visual area.
- Step 2. Return with goal accomplished.

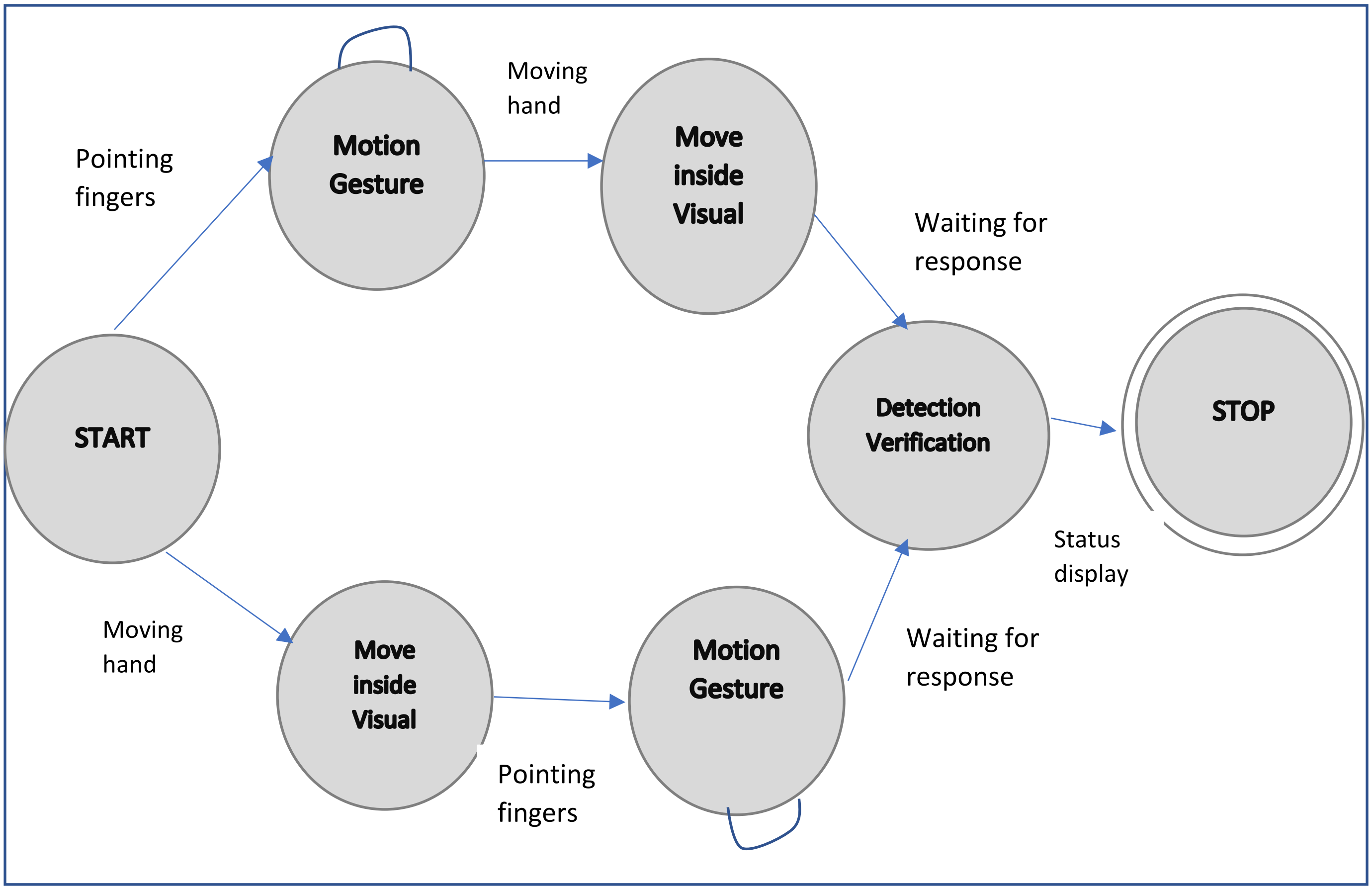
– Method for goal: Verifying the Recognition

- Step 1. Verify the Recognition.
- Step 2. Return with goal accomplished.

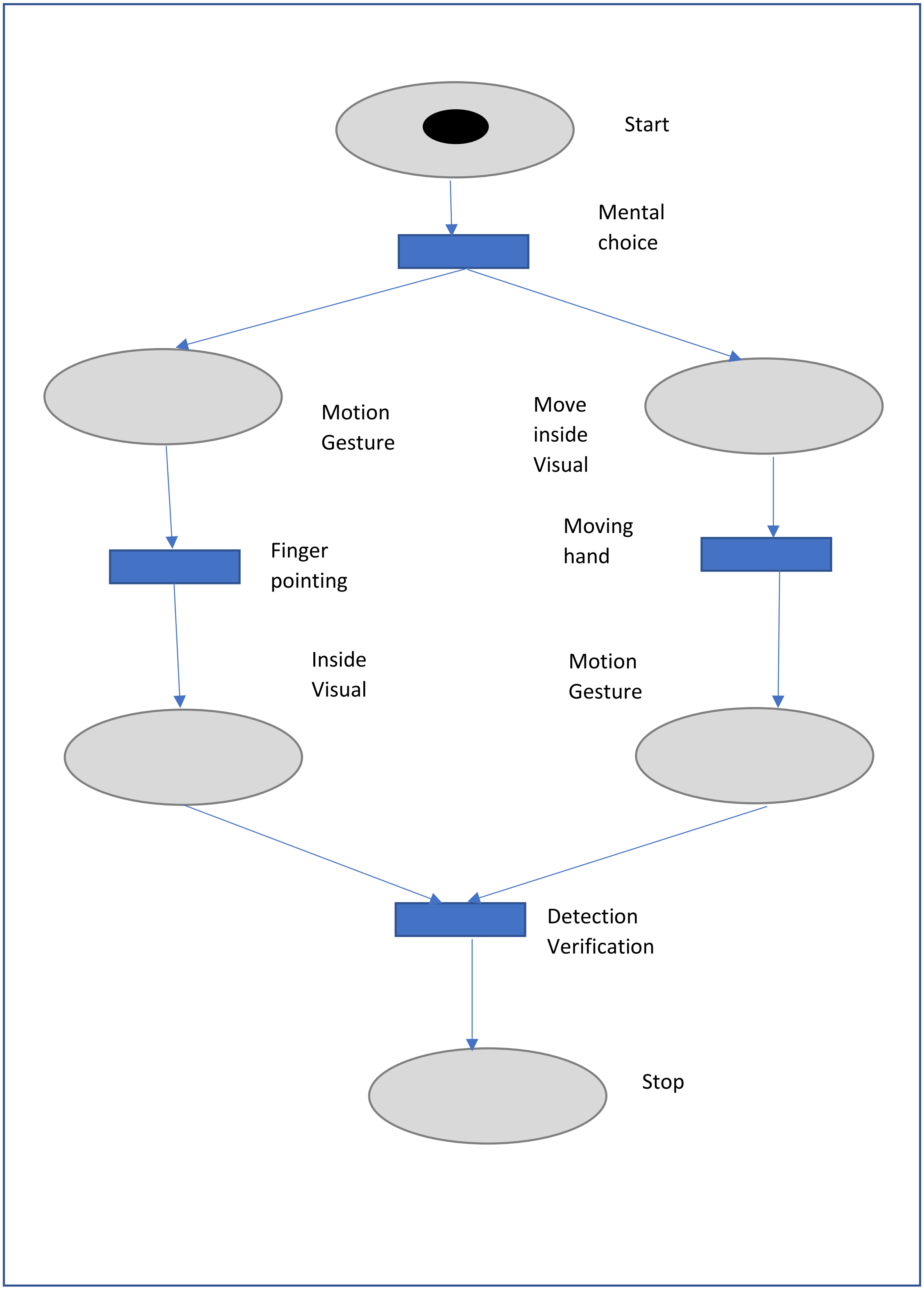
Comparison

	MHP	KLM
Movement outside the Visual Area	520ms	4.2
Movement Inside the visual Area	590ms	5.55

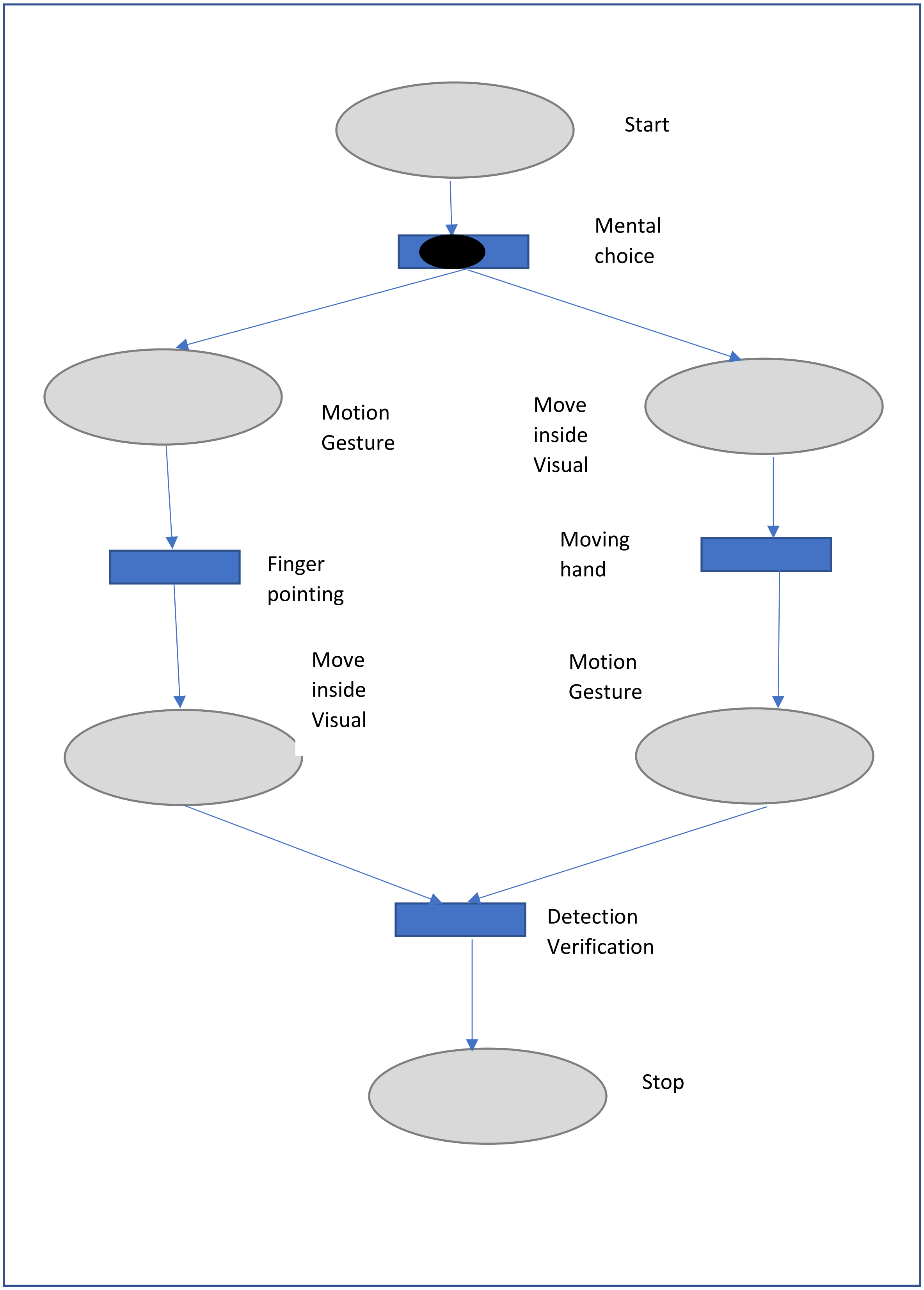
State Transition Network

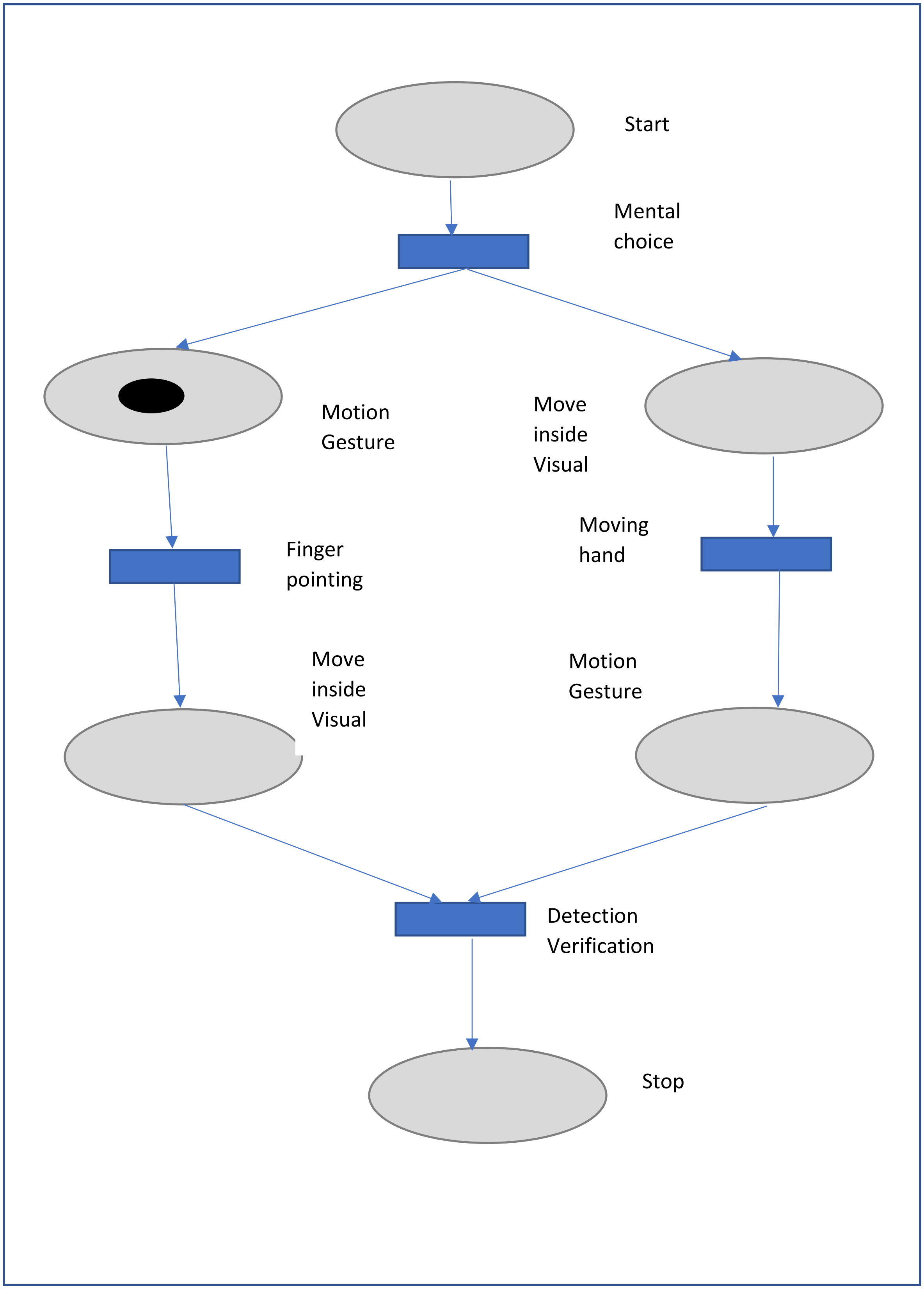


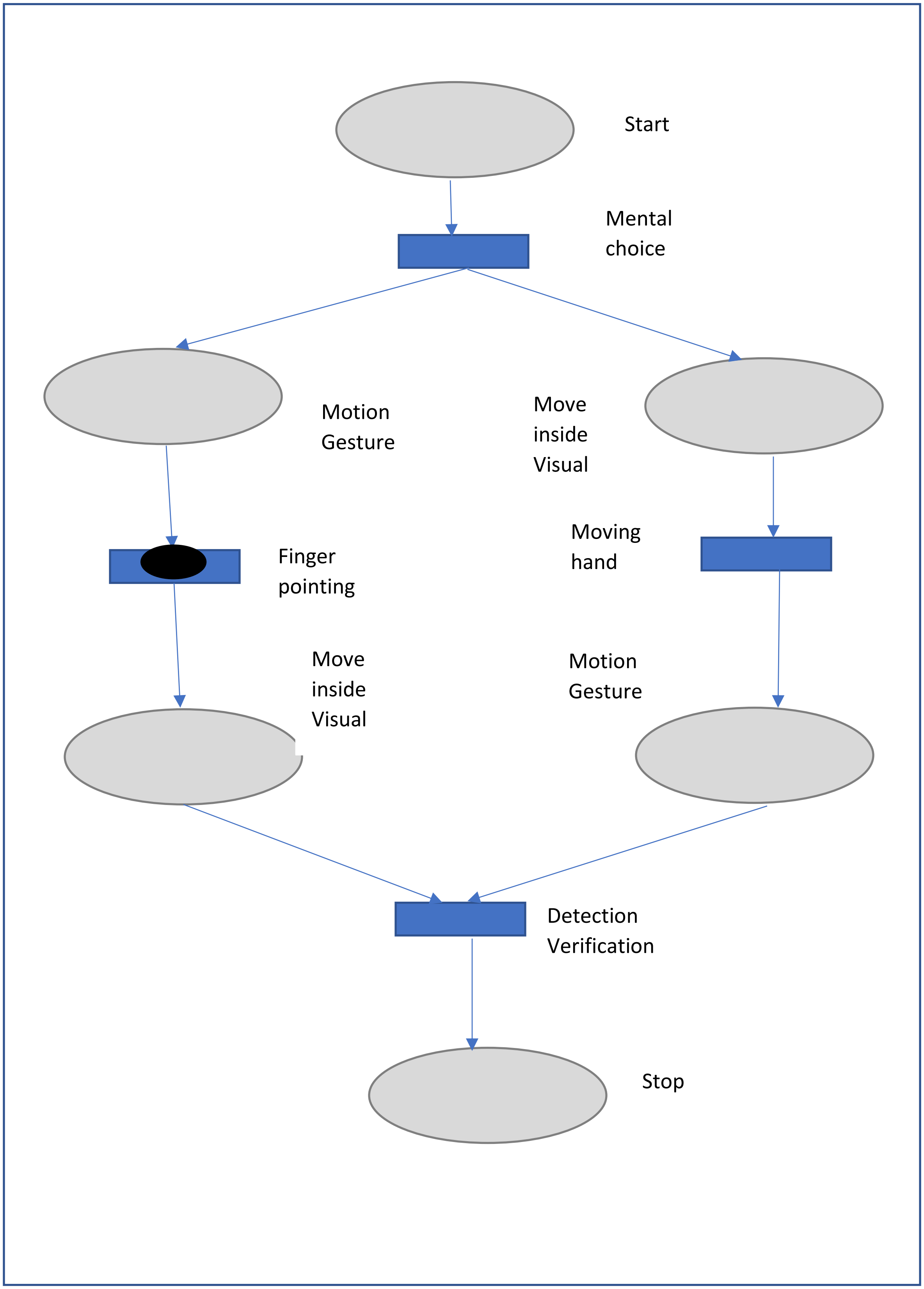
Petri-Mets

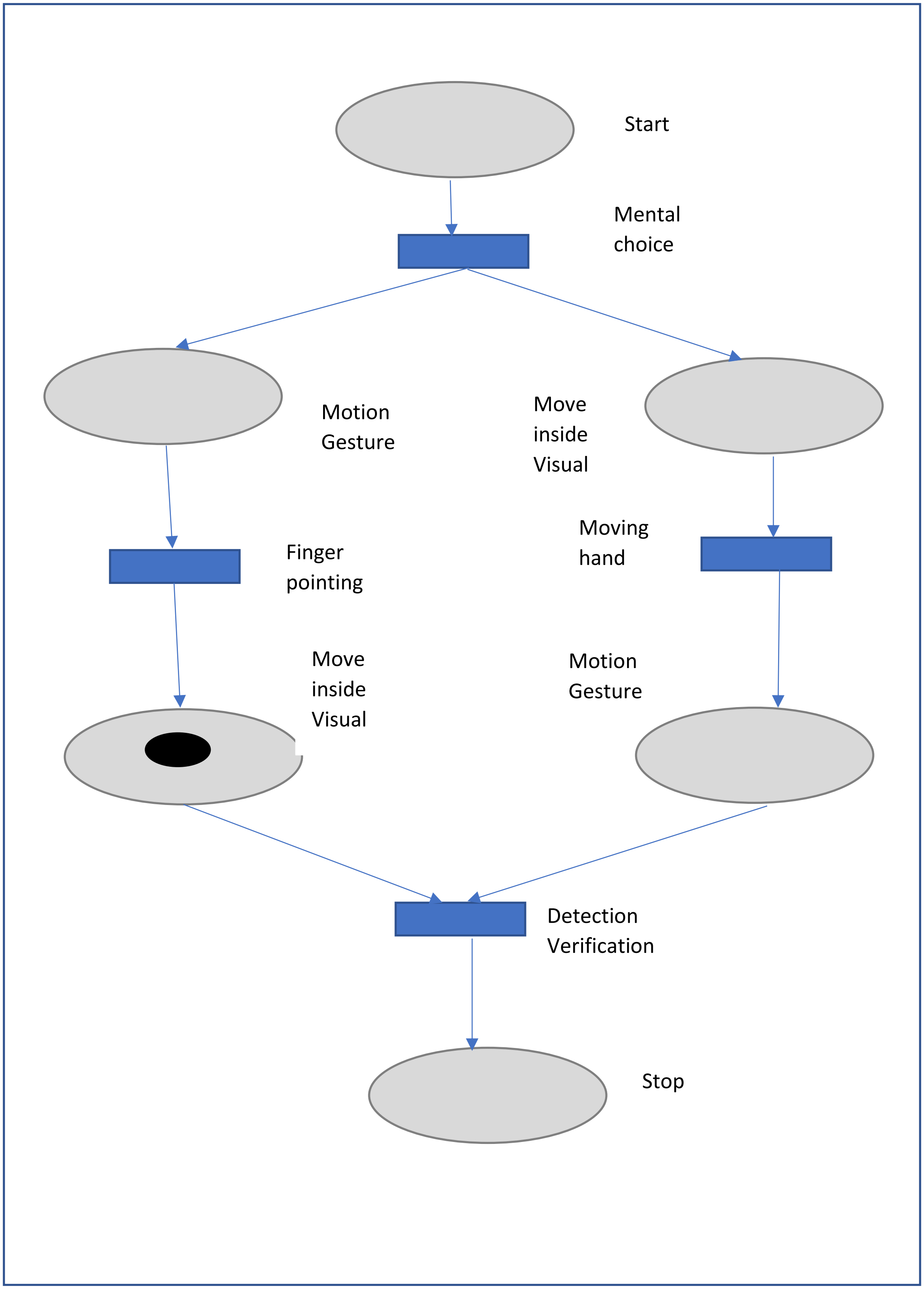


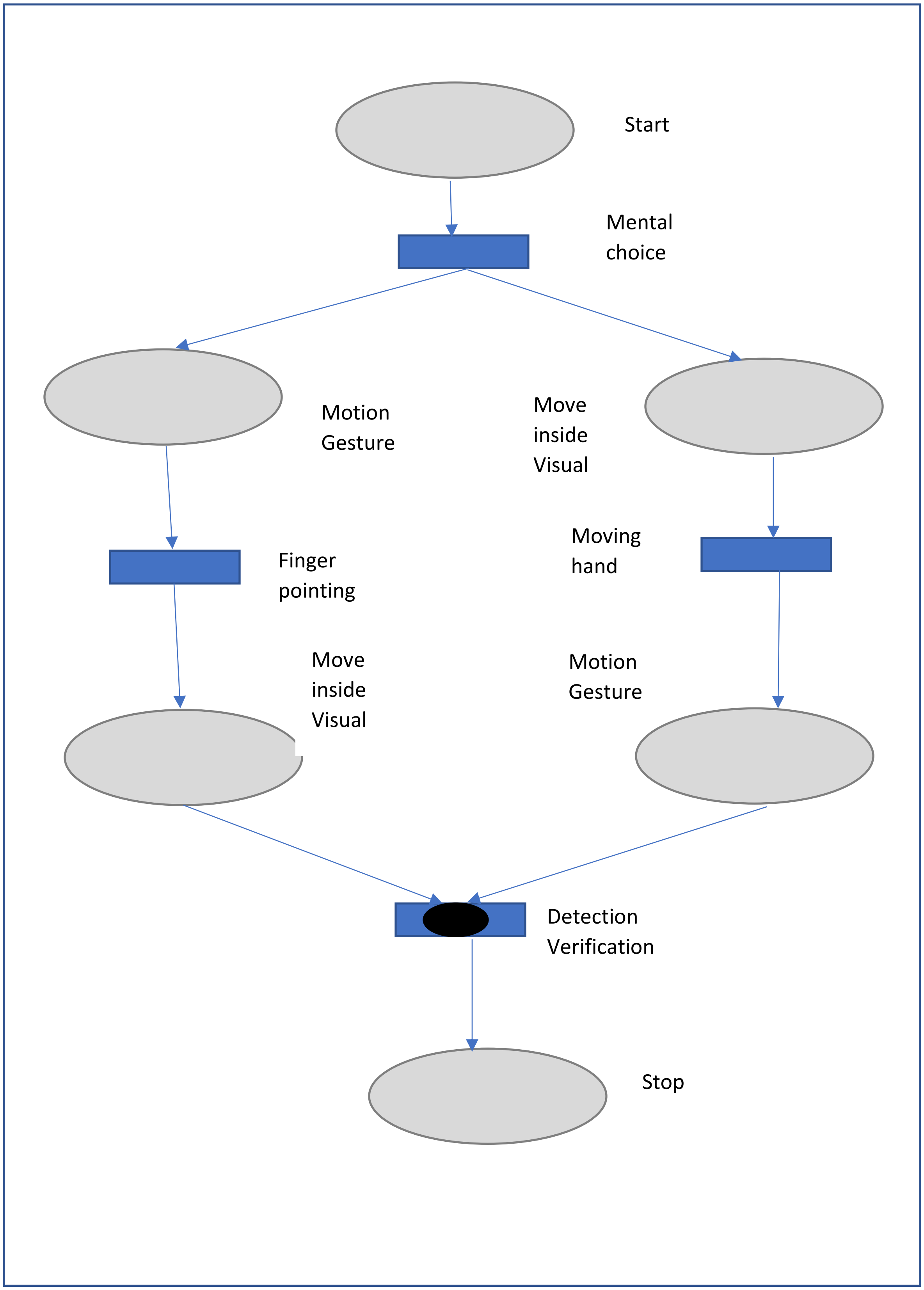
Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Inside Visual
Motion Gesture
Detection Verification
Stop

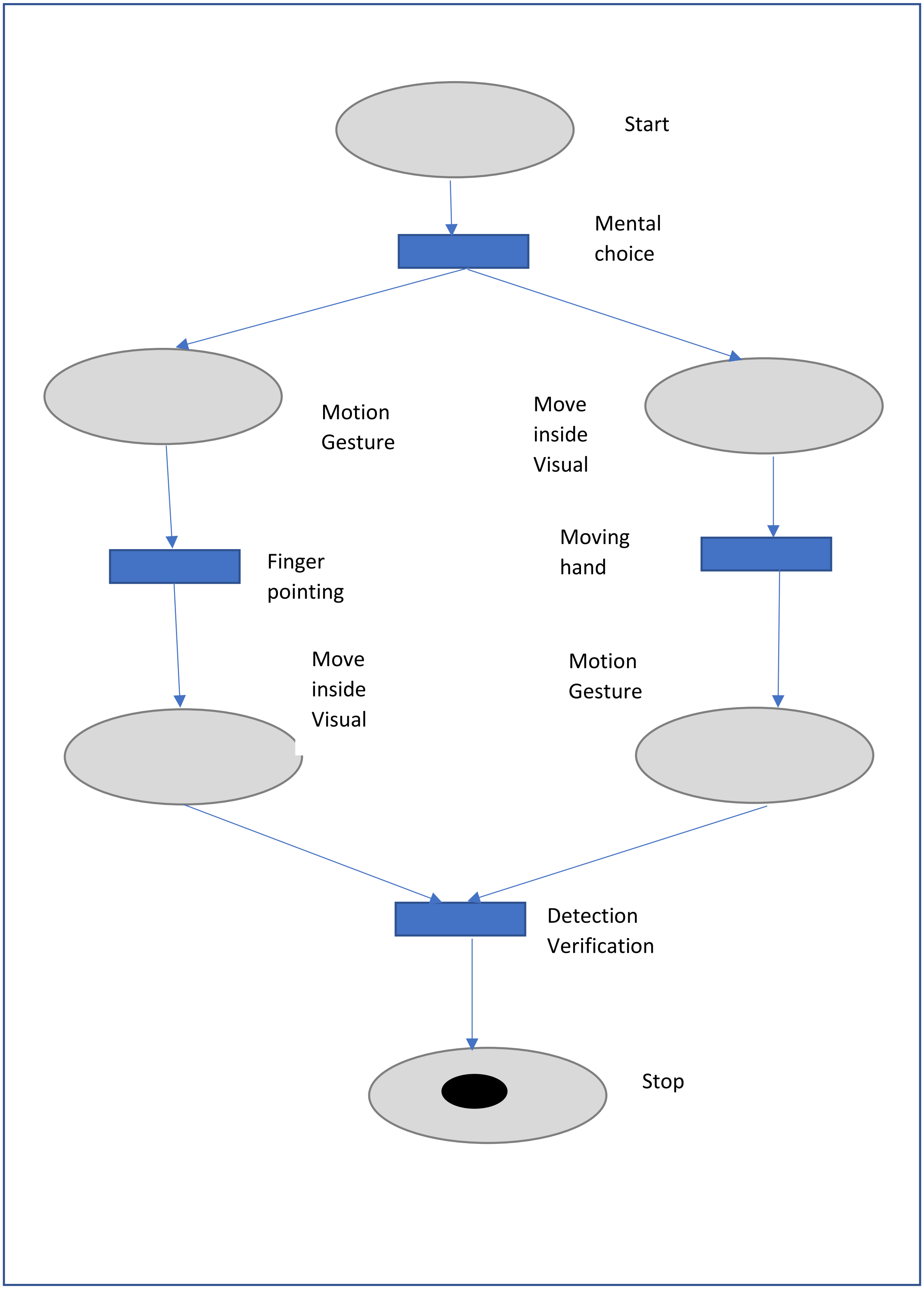

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

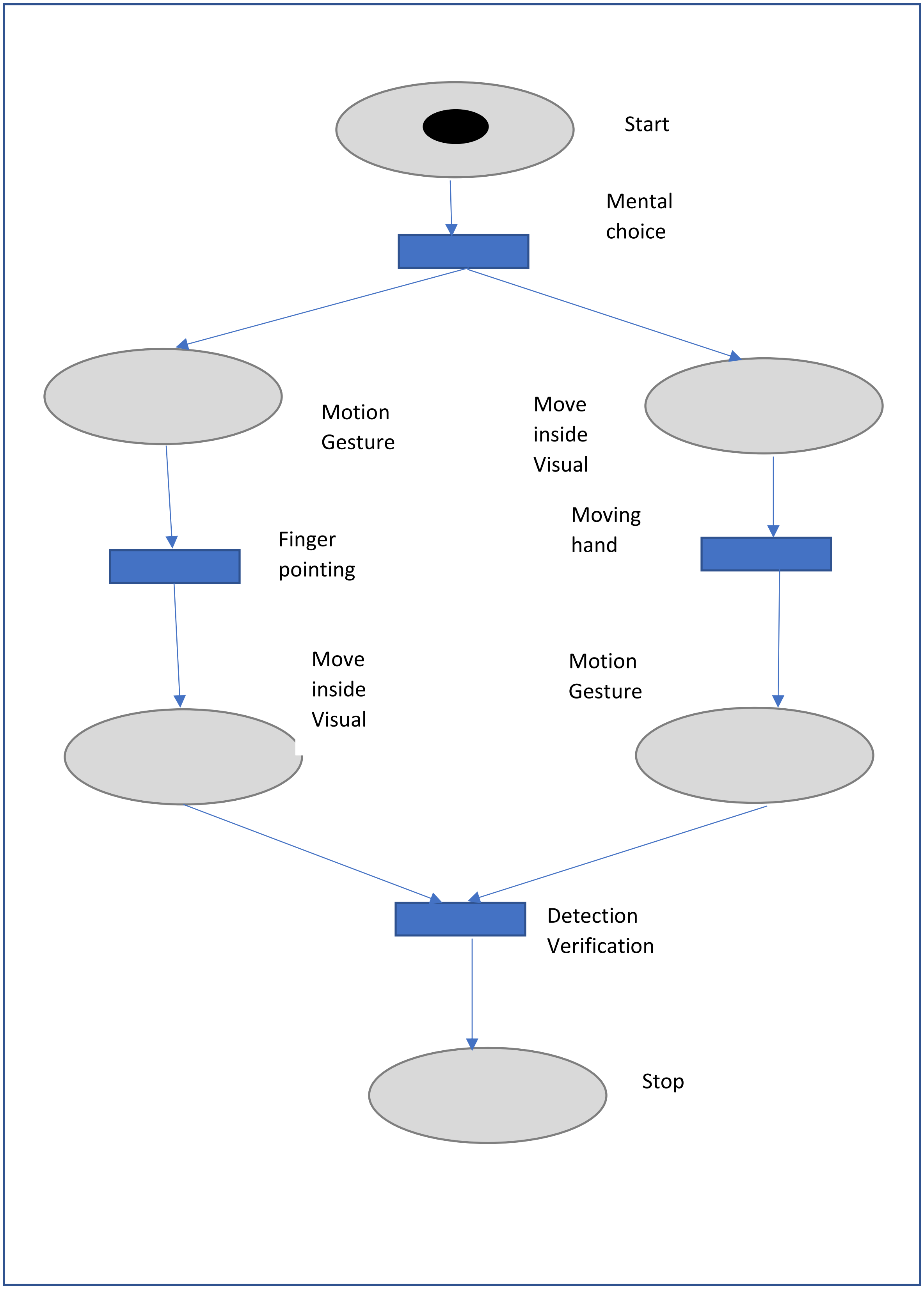

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

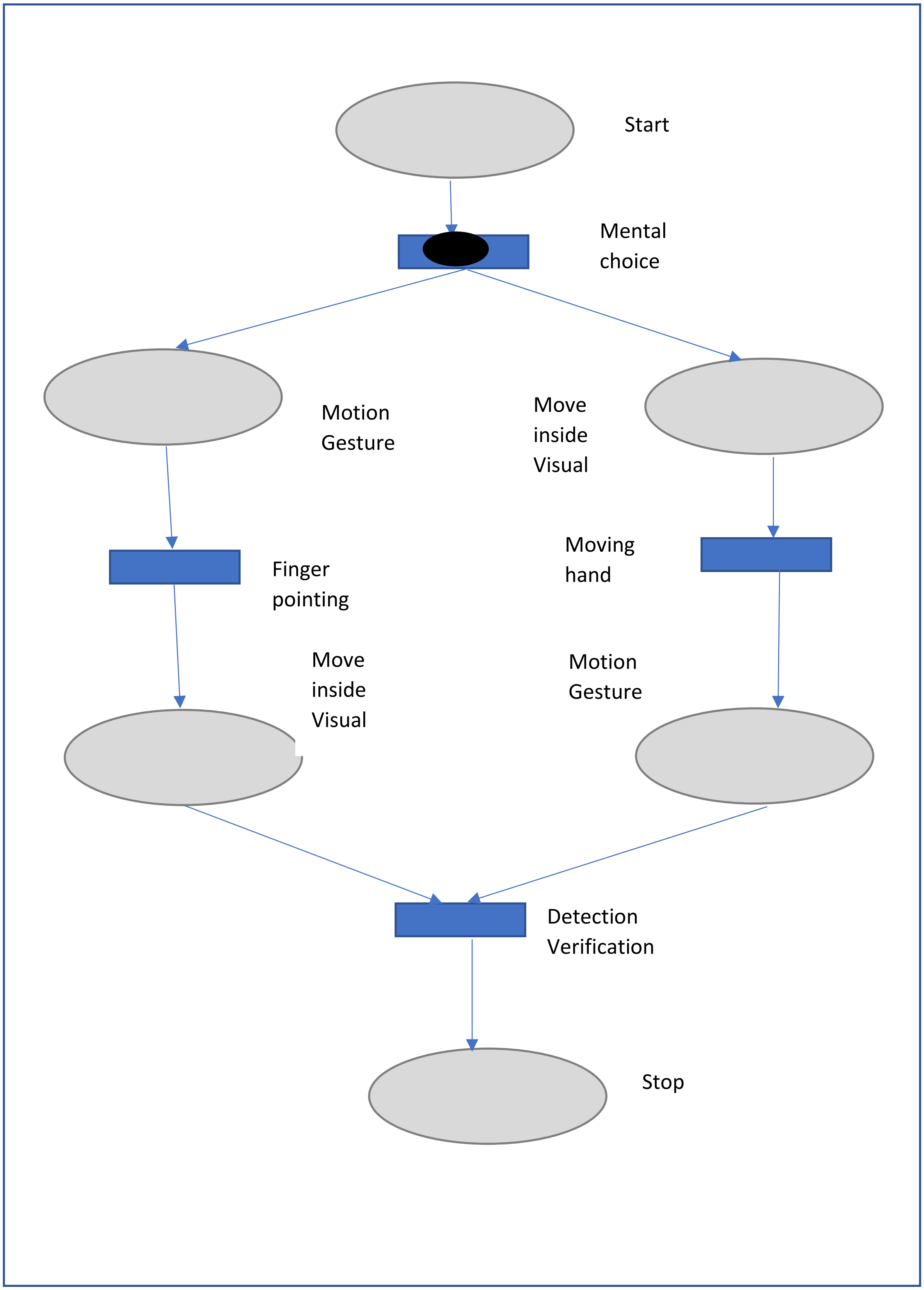

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

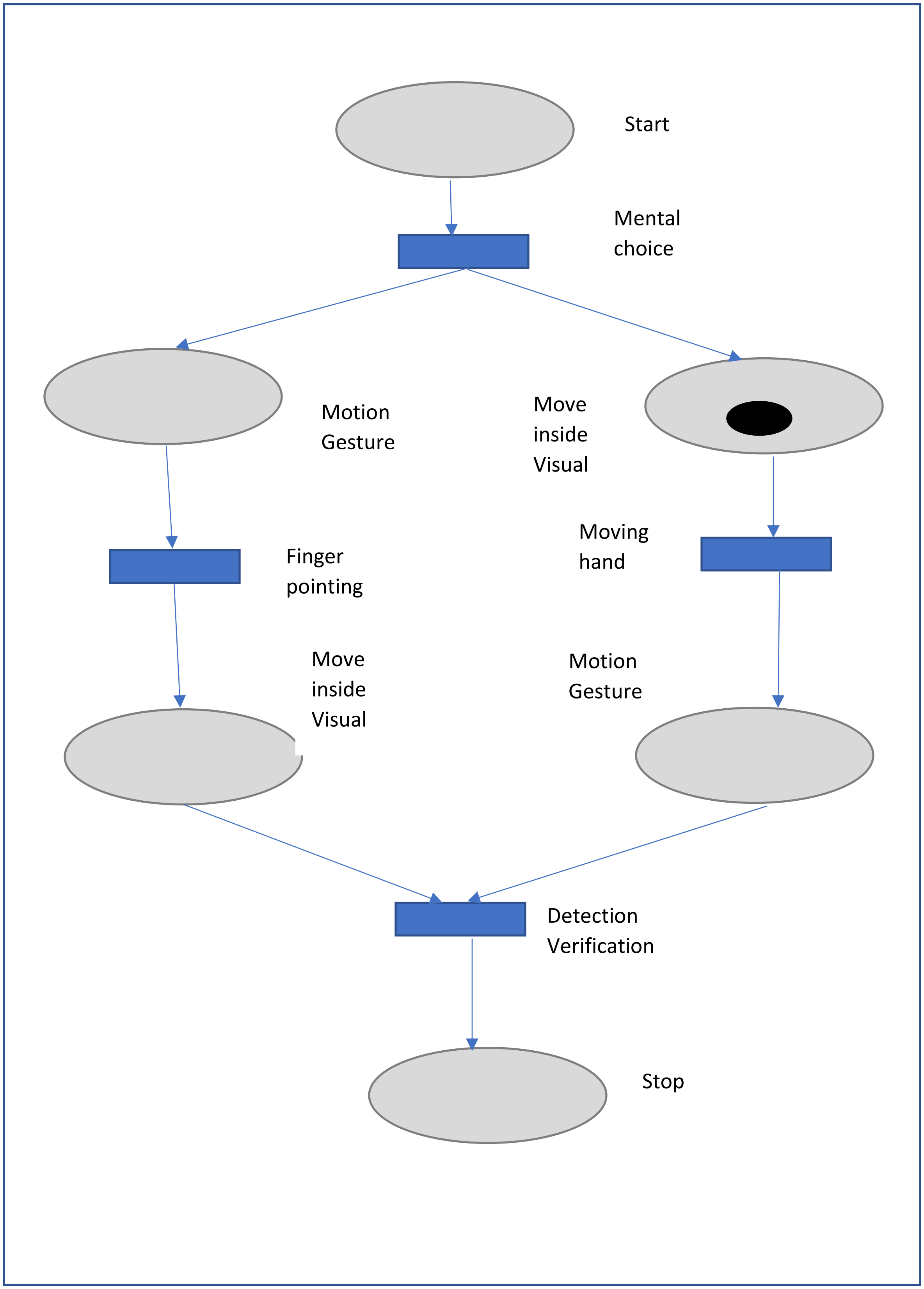

Start
Mental
choice
Motion
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Motion
Gesture
Detection
Verification
Stop

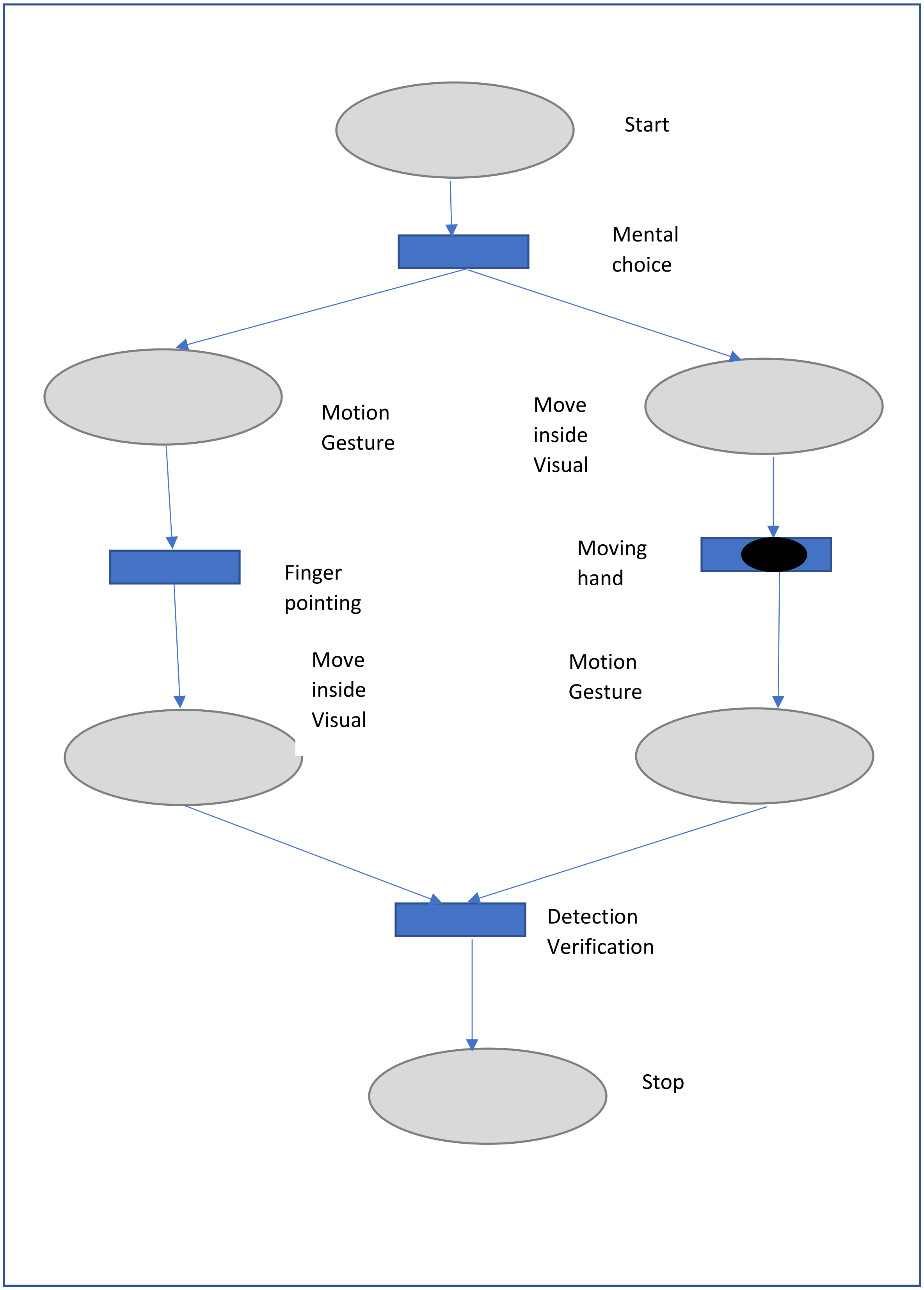

Start
Mental
choice
Motion
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Motion
Gesture
Detection
Verification
Stop

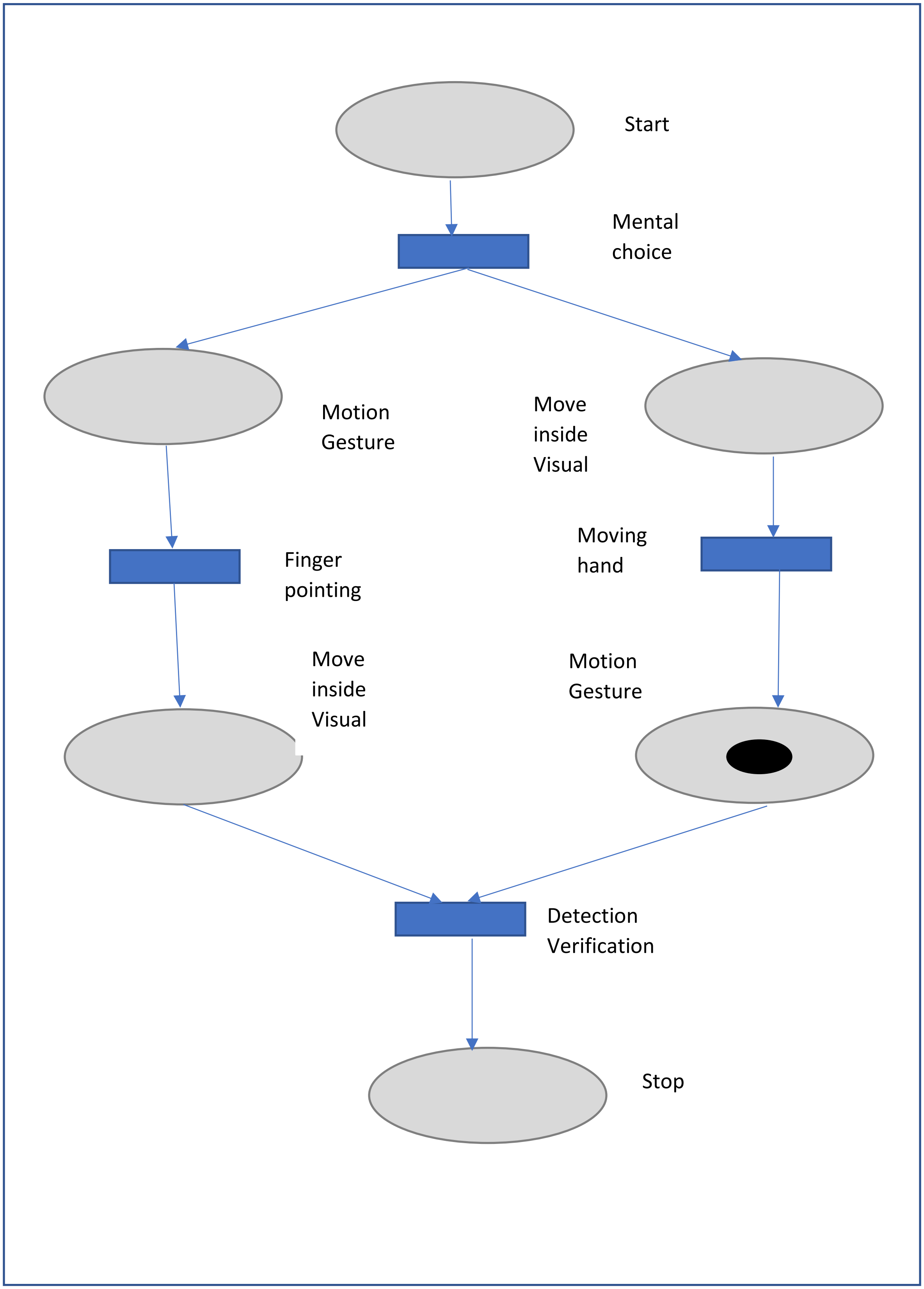

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop


Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

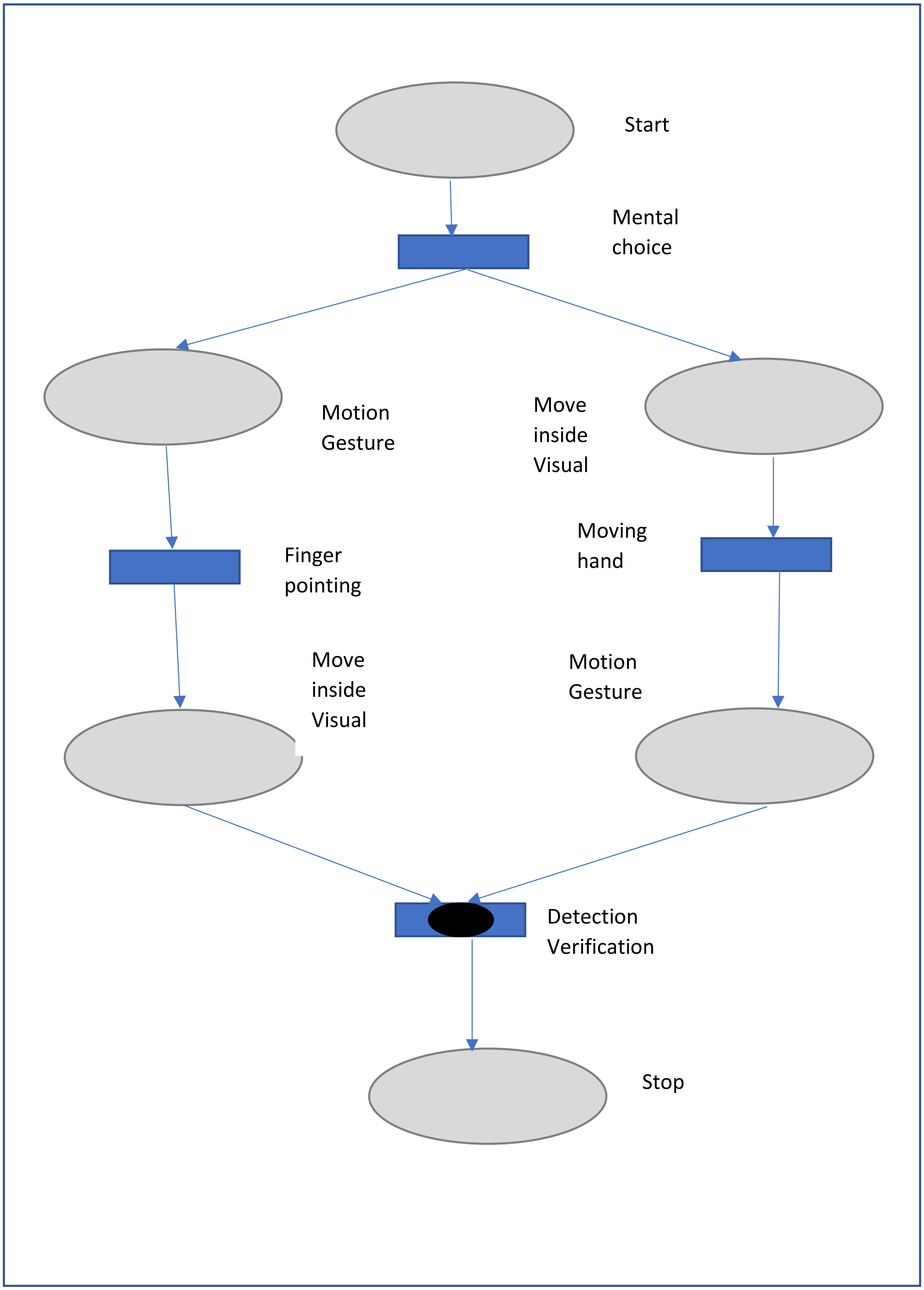

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

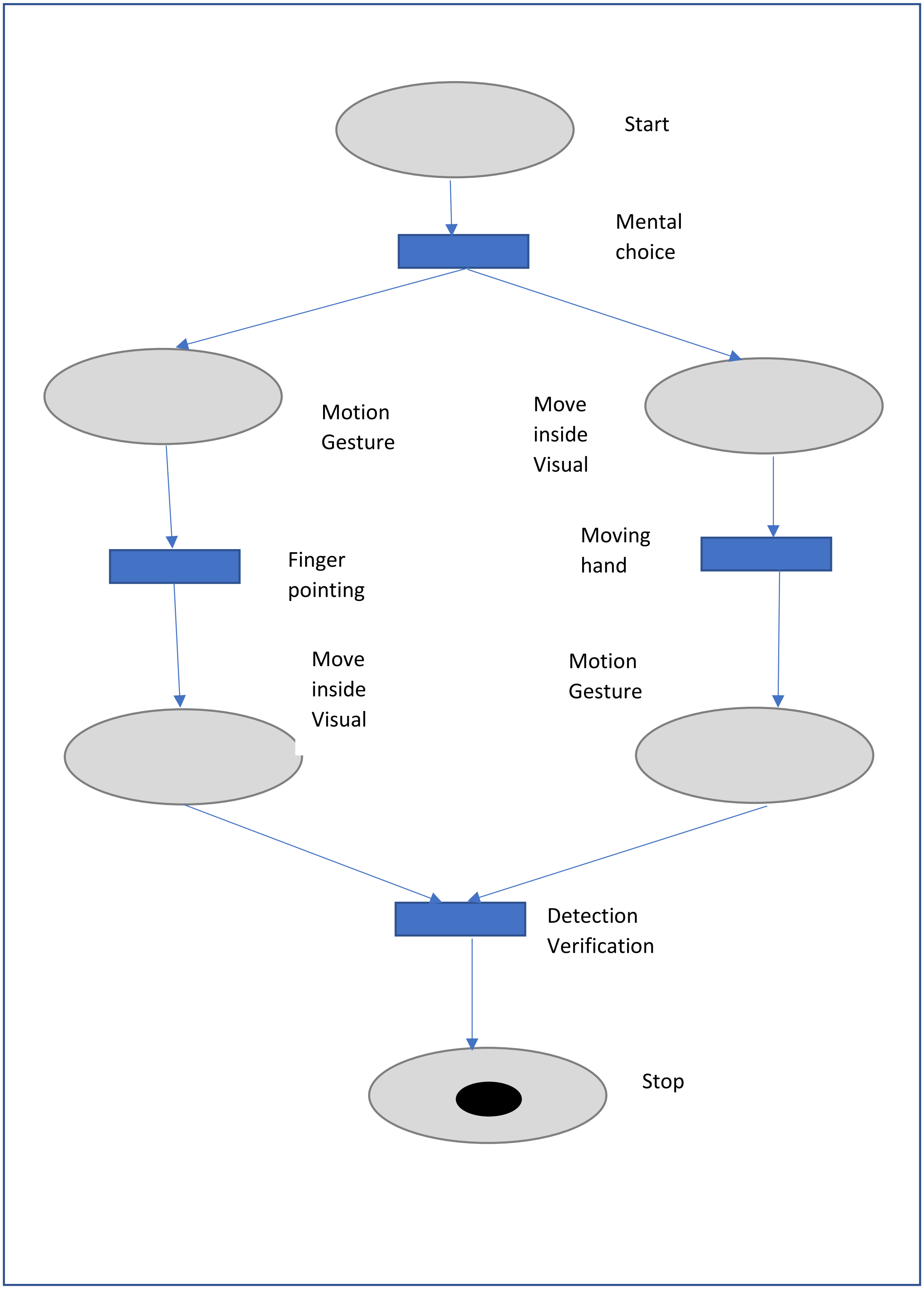

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop


Start
Mental
choice
Motion
Gesture
Move
inside
Visual
Finger
pointing
Moving
hand
Move
inside
Visual
Motion
Gesture
Detection
Verification
Stop

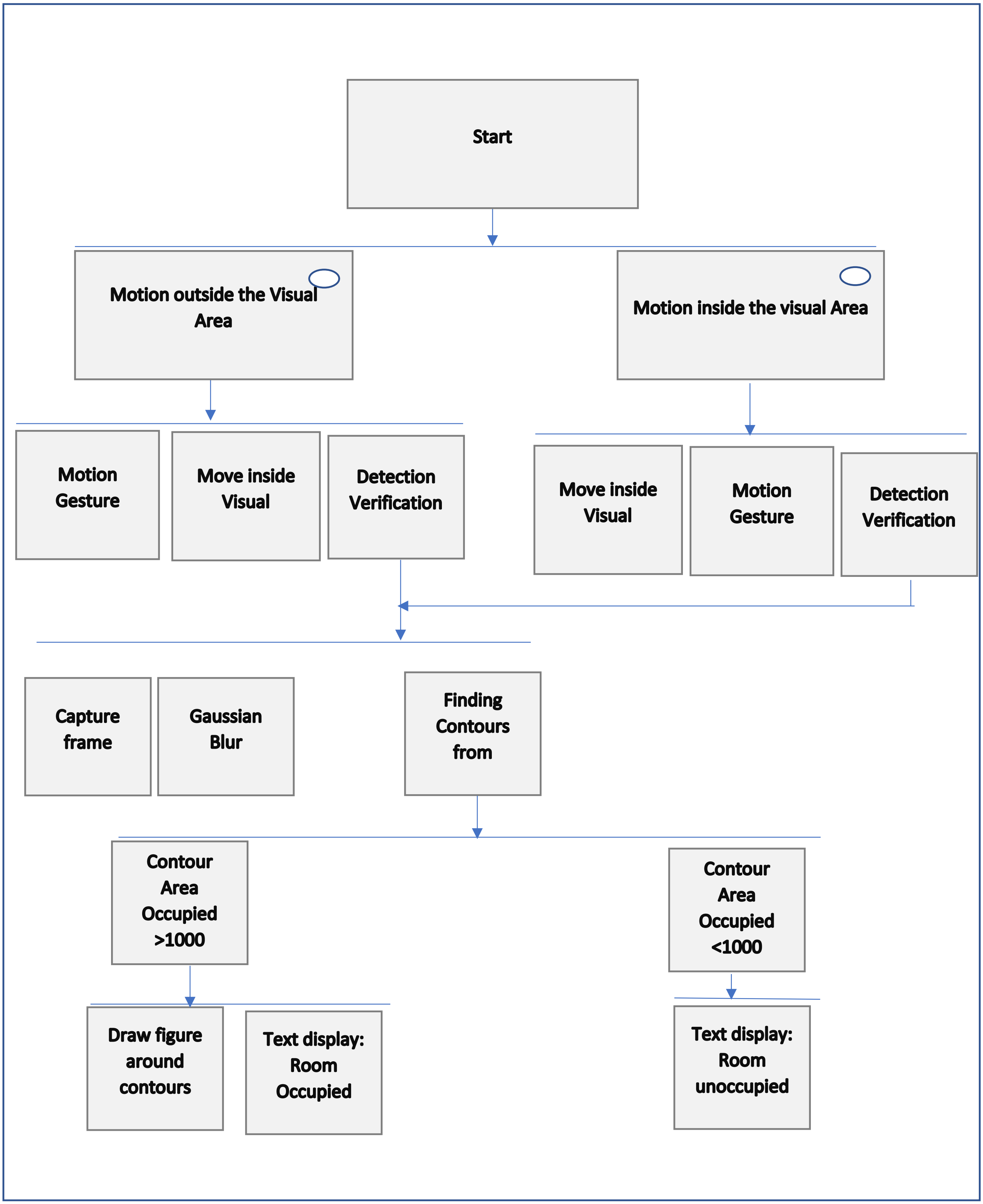


Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop


Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

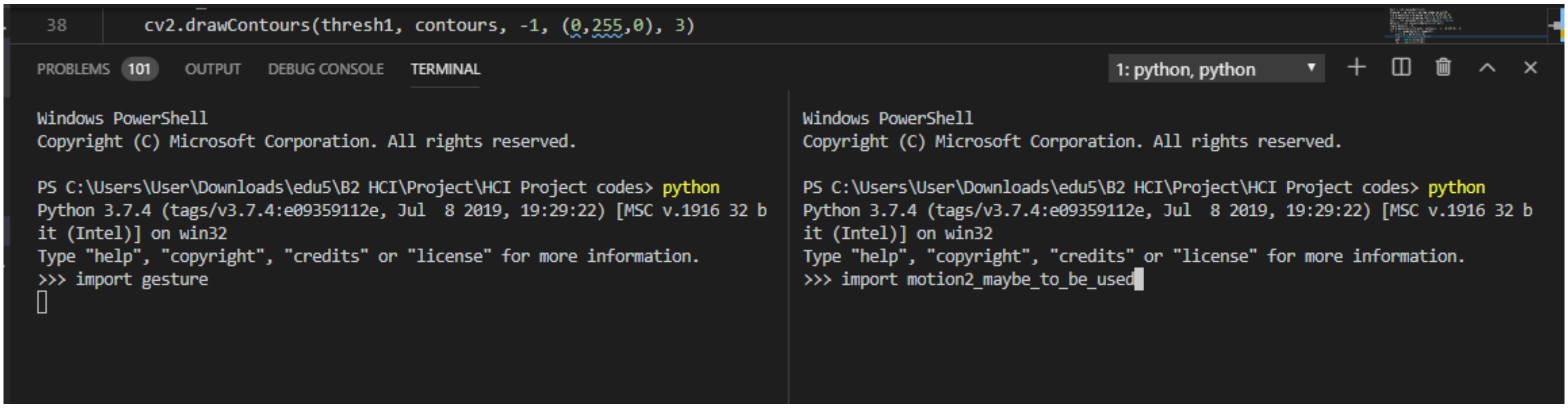

Start
Mental choice
Motion Gesture
Move inside Visual
Finger pointing
Moving hand
Move inside Visual
Motion Gesture
Detection Verification
Stop

Jackson Structured diagram



Review 3

Both the codes must run simultaneously in different terminals



Codes

```
#gesture
import cv2
import numpy as np
import math
import time
import datetime


cap = cv2.VideoCapture(0)
while(cap.isOpened()):
    ret, img = cap.read()
    cv2.rectangle(img,(300,300),(100,100),(0,255,0),0)
    crop_img = img[100:300, 100:300]
    grey = cv2.cvtColor(crop_img, cv2.COLOR_BGR2GRAY)
    value = (35, 35)
    blurred = cv2.GaussianBlur(grey, value, 0)
    _, thresh1 = cv2.threshold(blurred, 127, 255,cv2.THRESH_BINARY_INV
+cv2.THRESH_OTSU)
    cv2.imshow('Thresholded', thresh1)
```

```
    contours, hierarchy = cv2.findContours(thresh1.copy(),cv2.RETR_TRE
E, \
            cv2.CHAIN_APPROX_NONE)
    max_area = -1
    for i in range(len(contours)):
        cnt=contours[i]
        area = cv2.contourArea(cnt)
        if(area>max_area):
            max_area=area
            ci=i
    cnt=contours[ci]
    x,y,w,h = cv2.boundingRect(cnt)
    cv2.rectangle(crop_img,(x,y),(x+w,y+h),(0,0,255),0)
    hull = cv2.convexHull(cnt)
    drawing = np.zeros(crop_img.shape,np.uint8)
    cv2.drawContours(drawing,[cnt],0,(0,255,0),0)
    cv2.drawContours(drawing,[hull],0,(0,0,255),0)
    hull = cv2.convexHull(cnt,returnPoints = False)
    defects = cv2.convexityDefects(cnt,hull)
    count_defects = 0
    cv2.drawContours(thresh1, contours, -1, (0,255,0), 3)
    for i in range(defects.shape[0]):
        s,e,f,d = defects[i,0]
        start = tuple(cnt[s][0])
        end = tuple(cnt[e][0])
        far = tuple(cnt[f][0])
        a = math.sqrt((end[0] - start[0])**2 + (end[1] - start[1])**2)
        b = math.sqrt((far[0] - start[0])**2 + (far[1] - start[1])**2)
        c = math.sqrt((end[0] - far[0])**2 + (end[1] - far[1])**2)
        angle = math.acos((b**2 + c**2 - a**2)/(2*b*c)) * 57
```

```
        if angle <= 90:
            count_defects += 1
            cv2.circle(crop_img,far,1,[0,0,255],-1)
        dist = cv2.pointPolygonTest(cnt,far,True)
        cv2.line(crop_img,start,end,[0,255,0],2)
        cv2.circle(crop_img,far,5,[0,0,255],-1)
    if count_defects == 1:
        cv2.putText(img,"Fan speed 2", (50,50), cv2.FONT_HERSHEY_SIMPL
EX, 2, 2)
    elif count_defects == 2:
        cv2.putText(img, "Fan speed 3", (5,50), cv2.FONT_HERSHEY_SIMPL
EX, 1, 2)
    elif count_defects == 3:
        cv2.putText(img,"Fan speed 4", (50,50), cv2.FONT_HERSHEY_SIMPL
EX, 2, 2)
    elif count_defects == 4:
        cv2.putText(img,"fan speed 5", (50,50), cv2.FONT_HERSHEY_SIMPL
EX, 2, 2)
    else:
        cv2.putText(img,"Fan speed 1", (50,50), cv2.FONT_HERSHEY_SIMPL
EX, 2, 2)
    cv2.imshow('drawing', drawing)
    cv2.imshow('end', crop_img)
    cv2.imshow('Gesture', img)
    all_img = np.hstack((drawing, crop_img))
    cv2.imshow('Contours', all_img)
    k = cv2.waitKey(10)
    if k == 27:
        break
```

Code

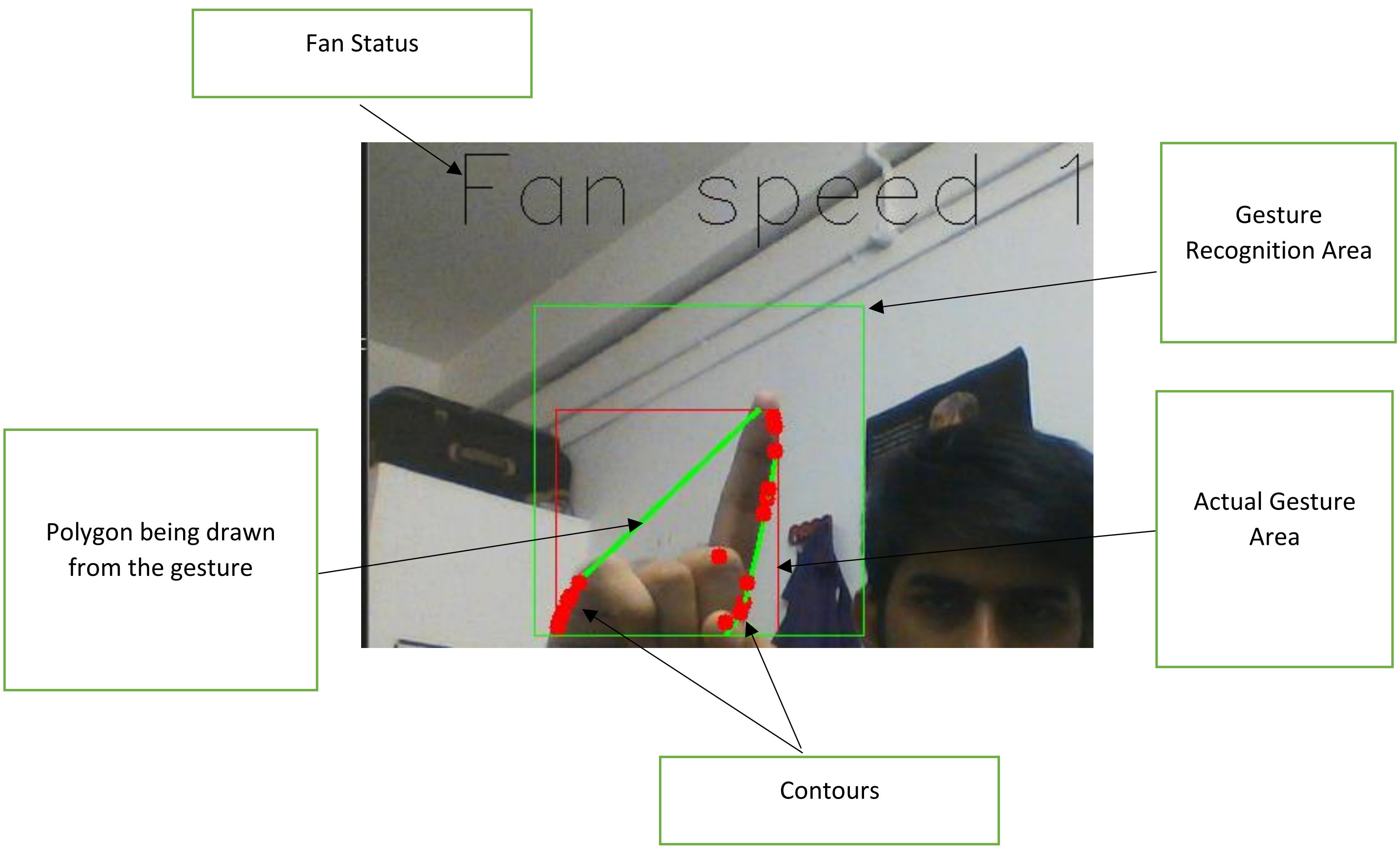
```
#motion2_maybe_to_be_used
import cv2
import numpy as np
import time
import datetime
def main():

   cap=cv2.VideoCapture(0)

   if cap.isOpened():

      ret,frame = cap.read()

   else:
      ret =False

   ret,frame1 = cap.read()
   ret,frame2 = cap.read()

   while ret:

      ret,frame = cap.read()

      text = 'Unoccupied'

      d=cv2.absdiff(frame1,frame2)

      grey=cv2.cvtColor(d,cv2.COLOR_BGR2GRAY)
```

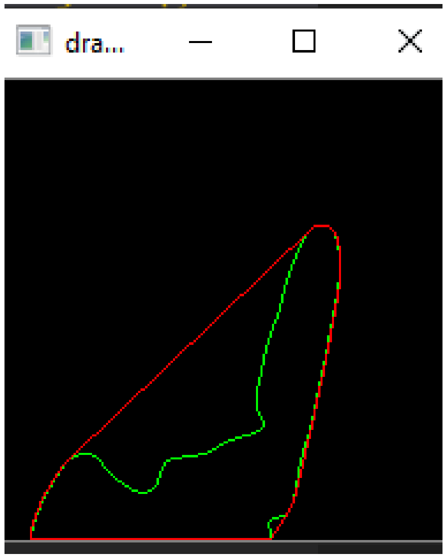
```
      blur =cv2.GaussianBlur(grey,(5,5),0)

      ret,th=cv2.threshold(blur,20,255,cv2.THRESH_BINARY)

      dilated=cv2.dilate(th,np.ones((3,3),np.uint8),iterations=3)

      cnt,h=cv2.findContours(dilated , cv2.RETR_TREE,cv2.CHAIN_APPROX_
SIMPLE)[-2:]


      for c in cnt:

         if cv2.contourArea(c) > 1000:

             (x, y, w, h) = cv2.boundingRect(c)


             cv2.rectangle(frame1, (x,y), (x+w, y+h), (0, 255, 0), 2)


             cv2.drawContours(frame1,c,-1,(0,255,0),2)


             text = 'Occupied'


         else:

             pass


         cv2.putText(frame1, '{+} Room Status: %s' % (text), (10,20),
cv2.FONT_HERSHEY_SIMPLEX , 0.5, (0, 0, 255), 2)


         cv2.putText(frame1, datetime.datetime.now().strftime('%A %d %
B %Y %I:%M:%S%p'), (10, frame.shape[0] - 10), cv2.FONT_HERSHEY_SIMPLEX
 , 0.35, (0, 0, 255),1)


      cv2.imshow("win1",frame2)

      cv2.imshow("inter",frame1)



      if cv2.waitKey(40) == 27:

         break
```

```
      frame1 = frame2
      ret,frame2 = cap.read()
      key = cv2.waitKey(1) & 0xFF
      if key == ord('q'):
        cv2.destroyAllWindows()
        break

main()
```

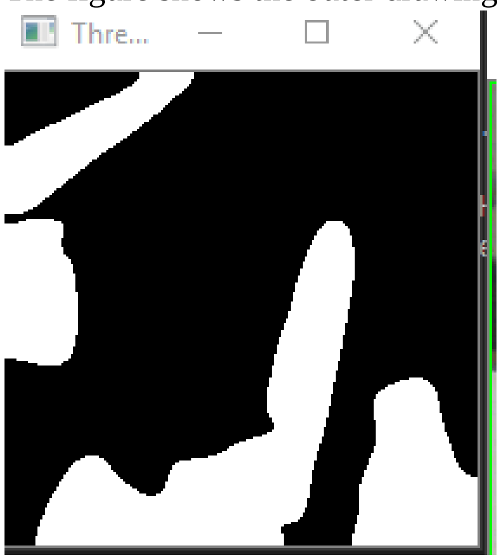
Outputs

1. Gesture Recognition

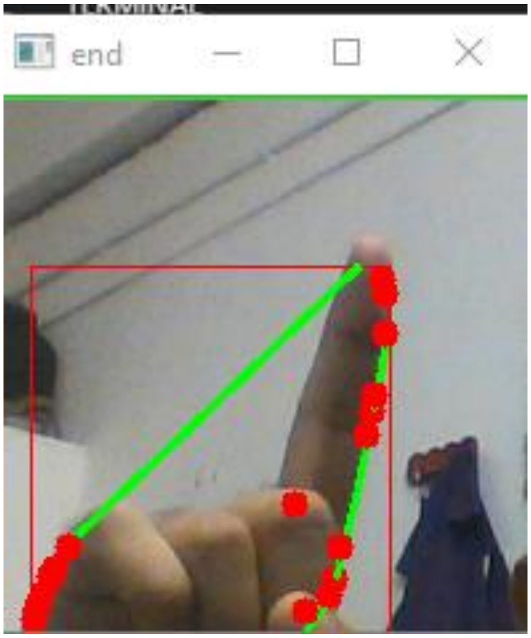




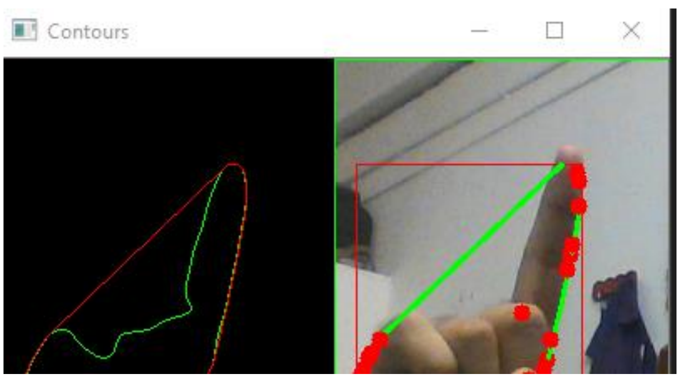
The figure shows the outer drawing of the finger pointing 1.



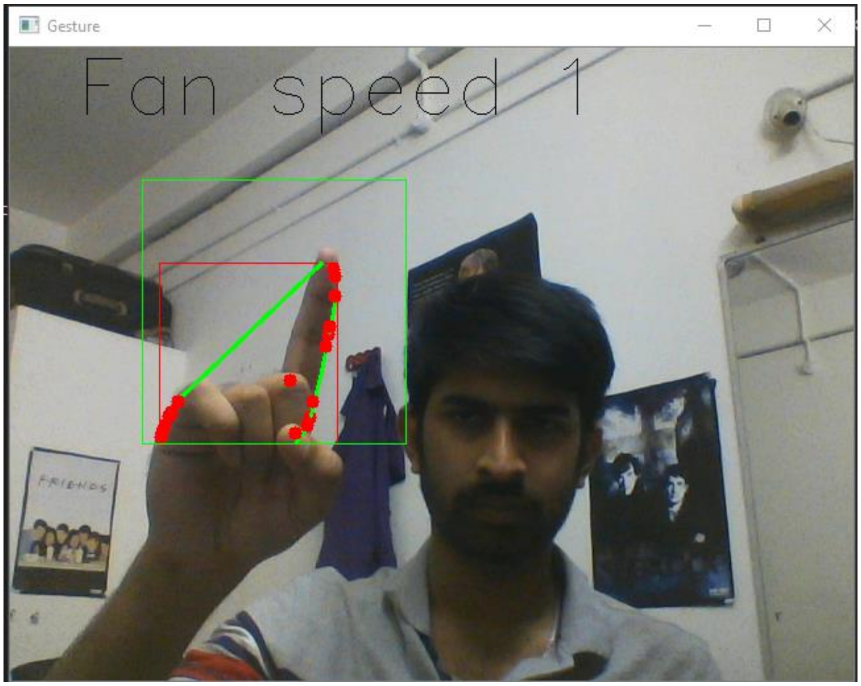
The figure shows the otus thresholding of the finger pointing 1.



The figure shows the end points of the gesture.



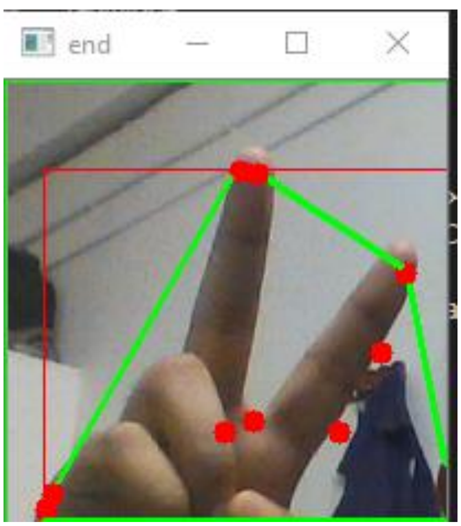
The figure shows the number of contours of the finger pointing 1.



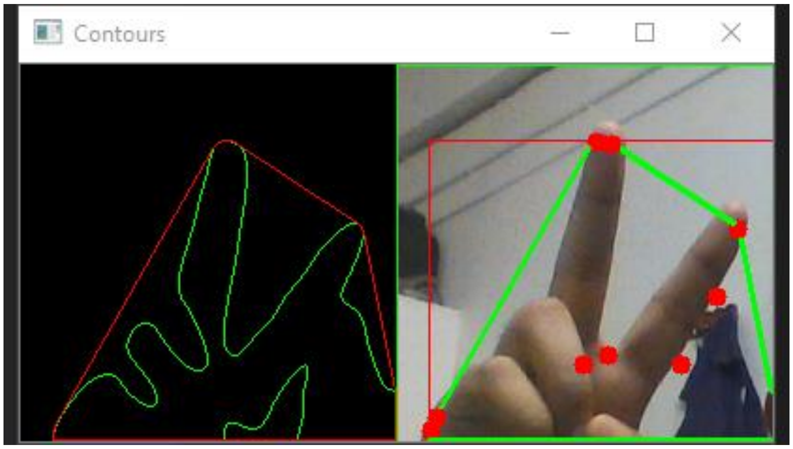
Fan speed 1 is triggered from the gesture pointing 1 in the box.

The figure shows the outer drawing of the finger pointing 2.

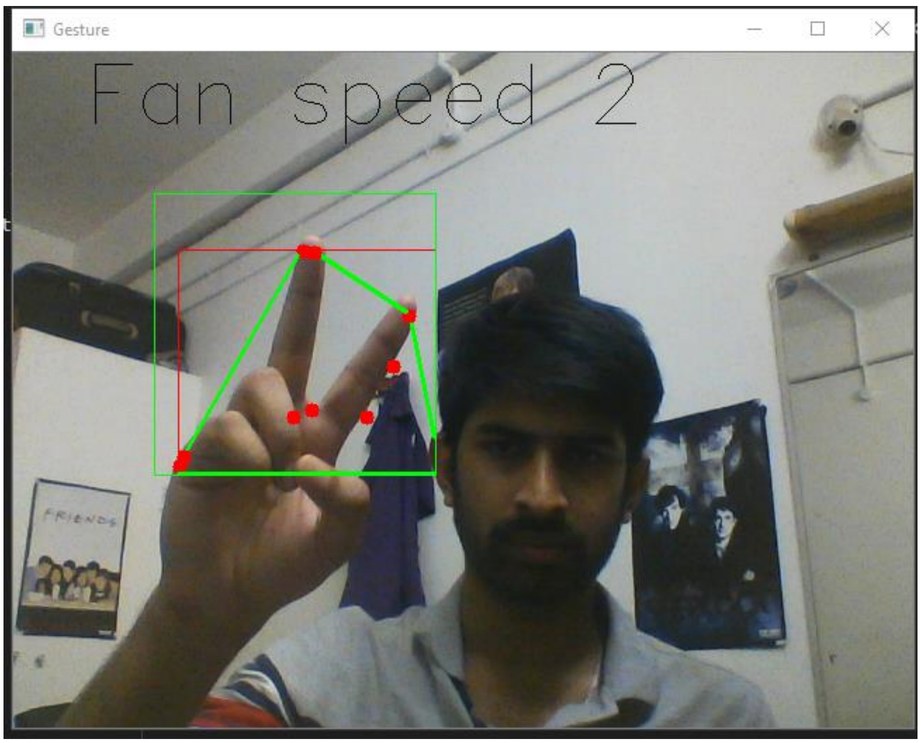
The figure shows the otus thresholding of the finger pointing 2.



The figure shows the end points of the gesture.

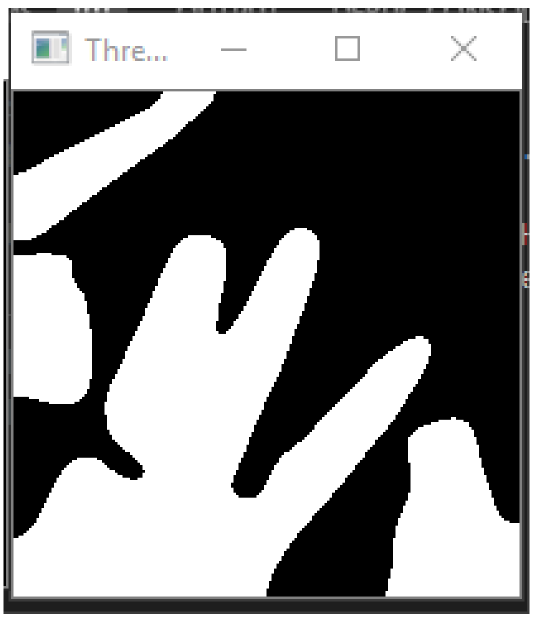


The figure shows the number of contours of the finger pointing 2.

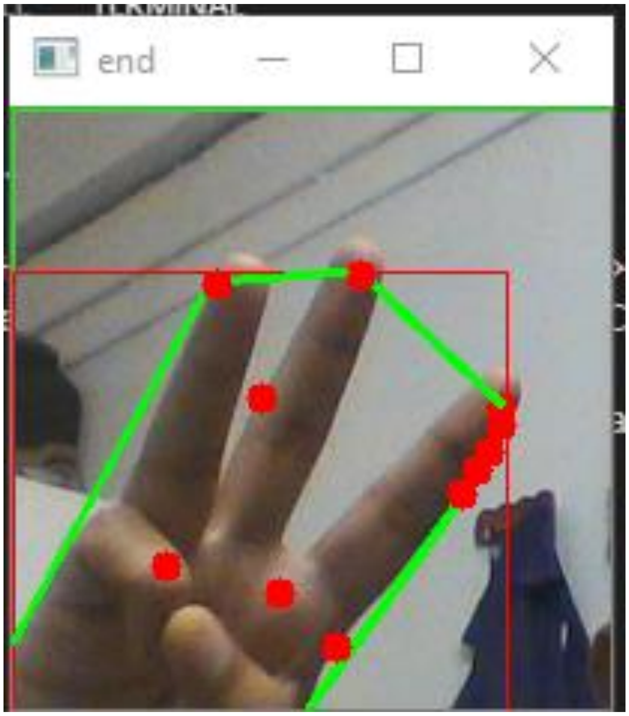


Fan speed 2 is triggered from the gesture pointing 2 in the box.

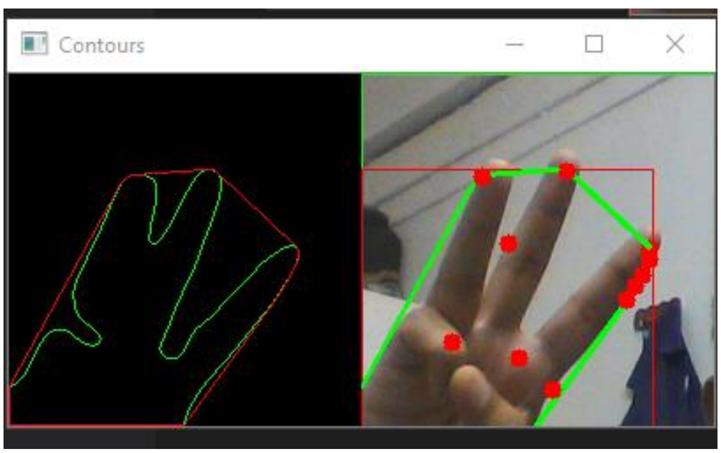
The figure shows the outer drawing of the finger pointing 3.



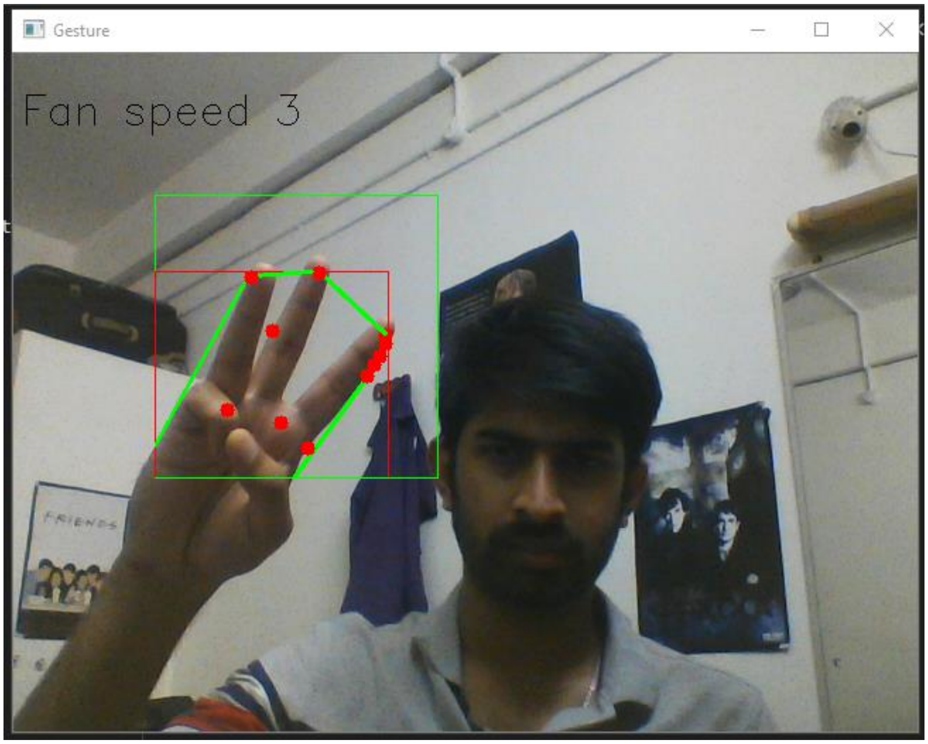
The figure shows the otus thresholding of the finger pointing 3.



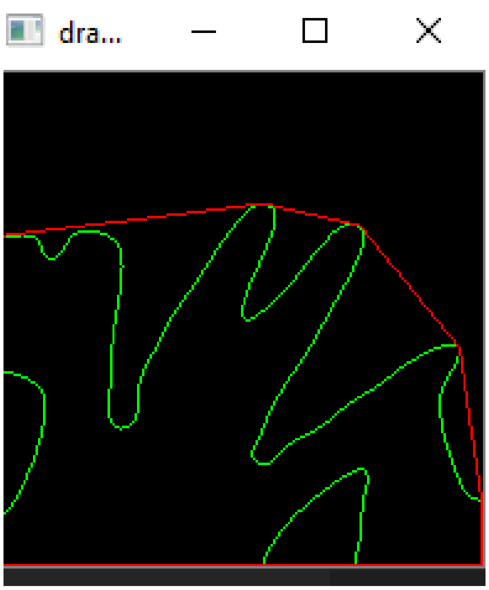
The figure shows the end points of the gesture.



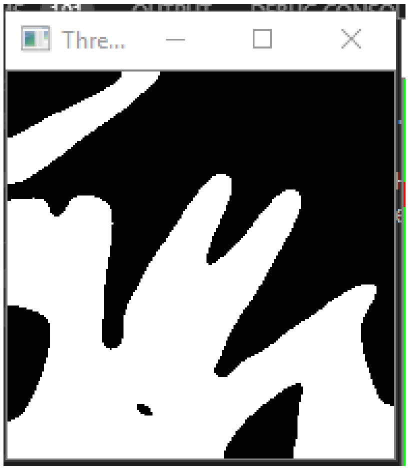
The figure shows the number of contours of the finger pointing 3.



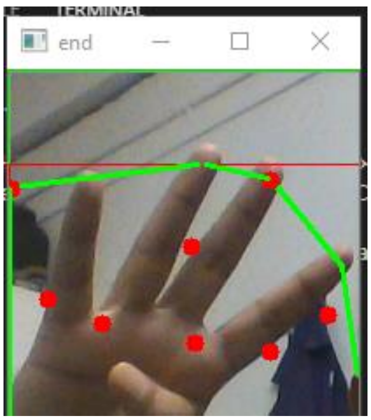
Fan speed 3 is triggered from the gesture pointing 3 in the box.



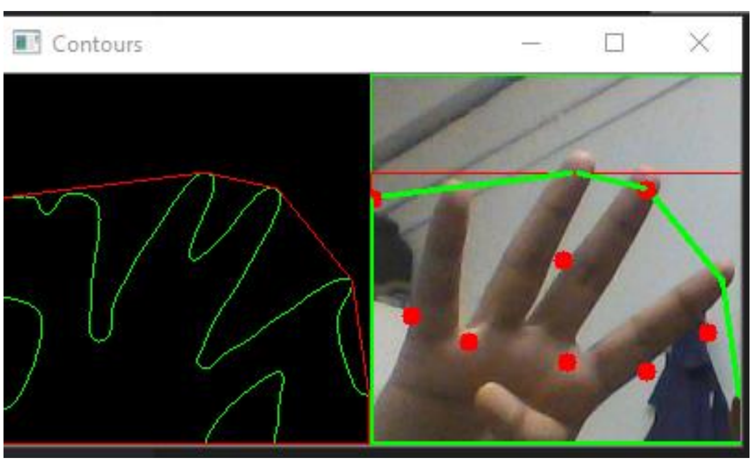
The figure shows the outer drawing of the finger pointing 4.



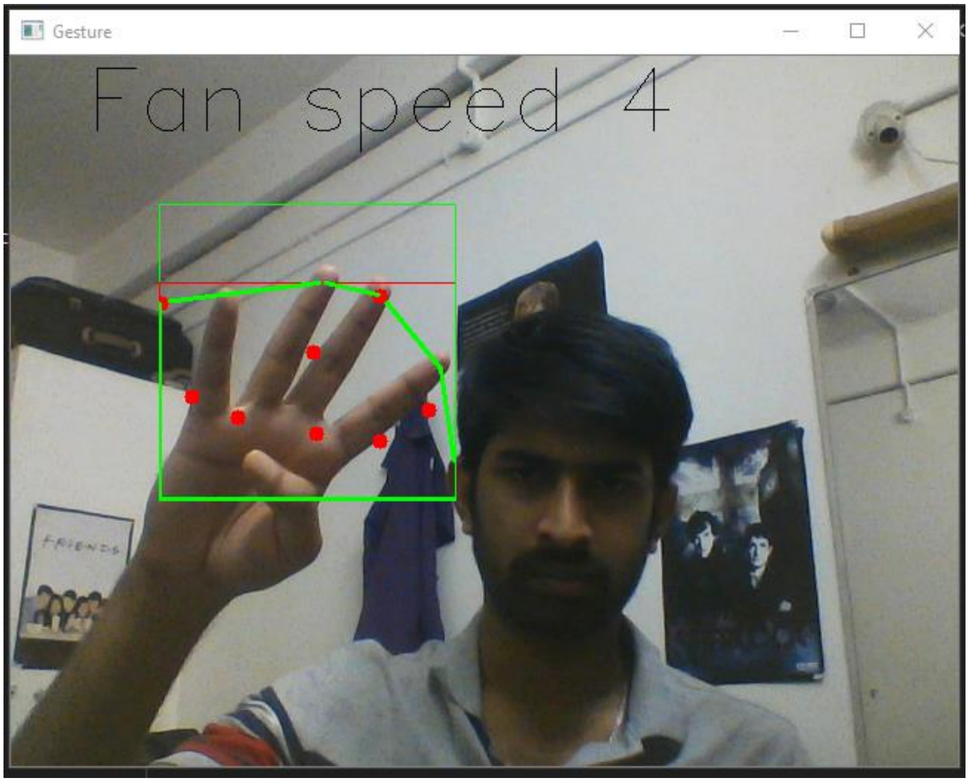
The figure shows the otus thresholding of the finger pointing 4.



The figure shows the end points of the gesture.



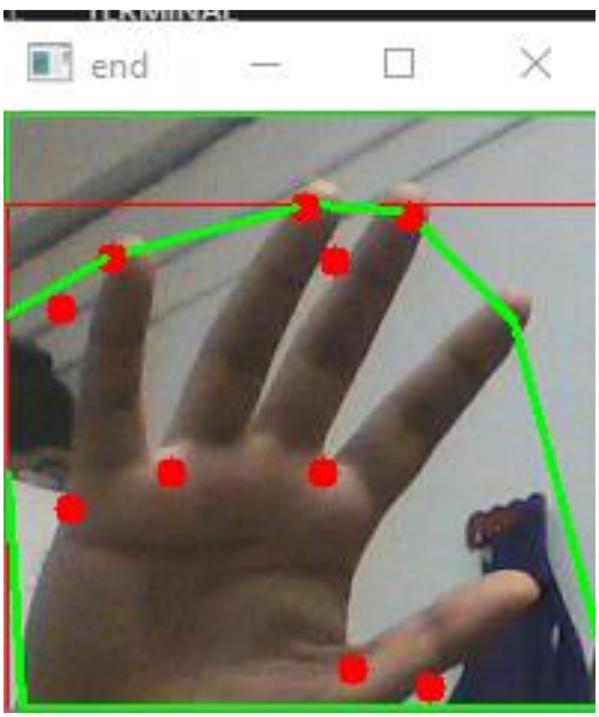
The figure shows the number of contours of the finger pointing 4.



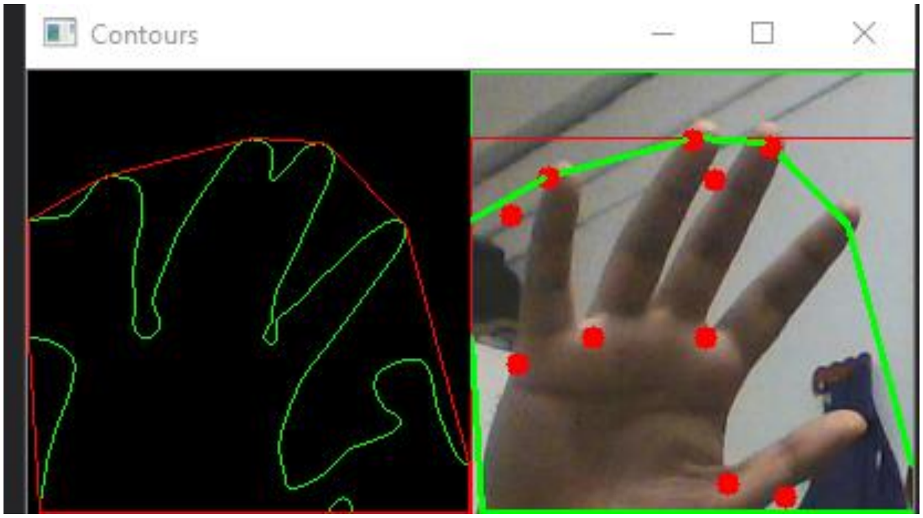
Fan speed 3 is triggered from the gesture pointing 3 in the box.

The figure shows the outer drawing of the finger pointing 5.

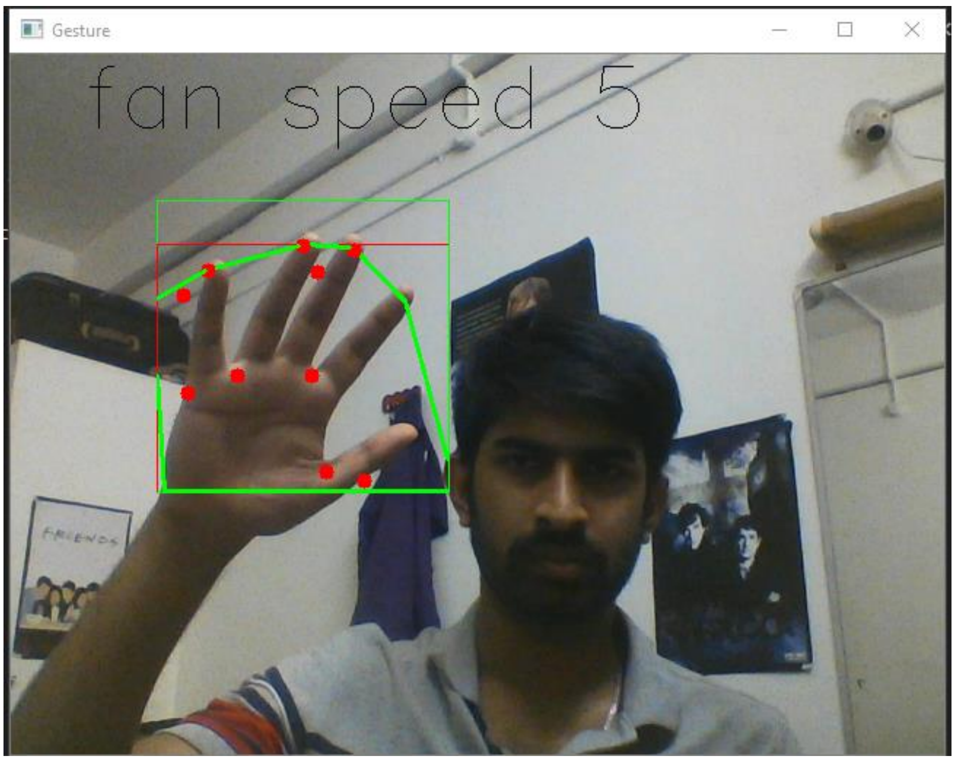
The figure shows the otus thresholding of the finger pointing 5.



The figure shows the end points of the gesture.

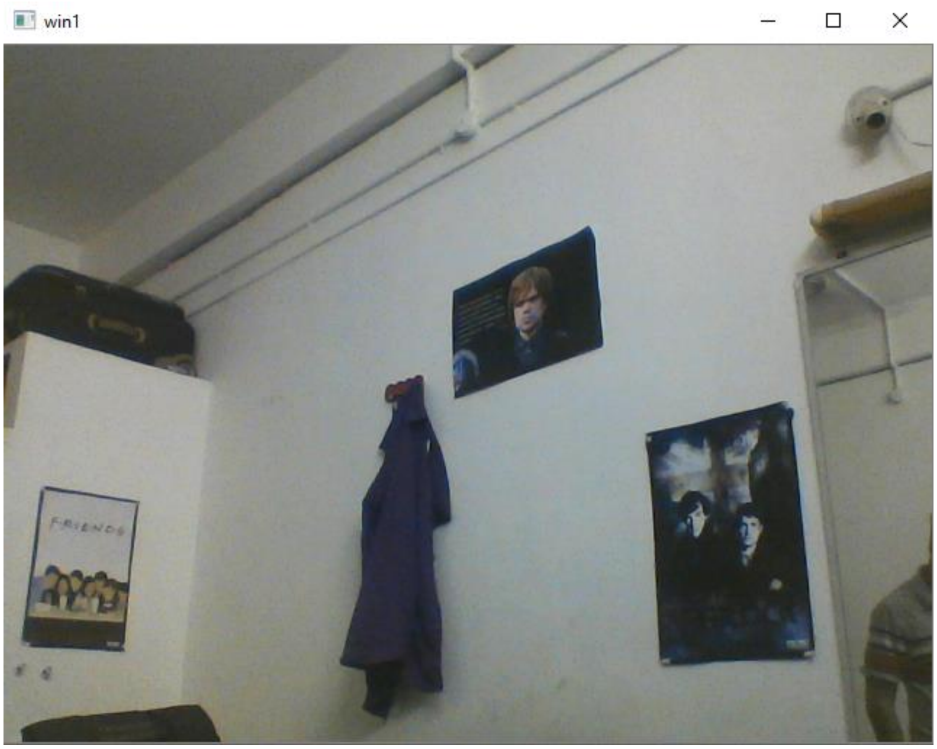


The figure shows the number of contours of the finger pointing 5.

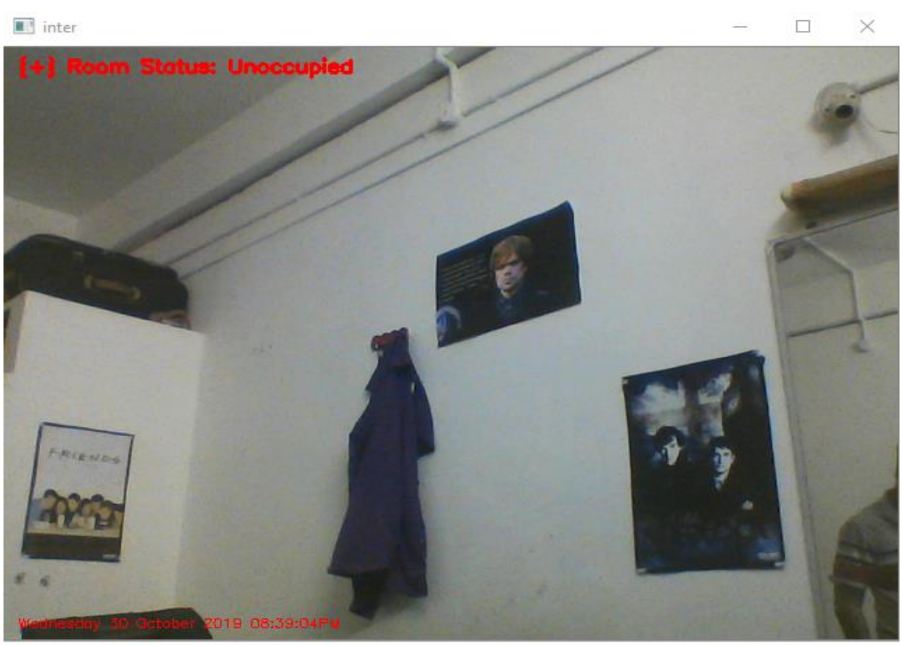


Fan speed 5 is triggered from the gesture pointing 5 in the box.

2. Motion detection

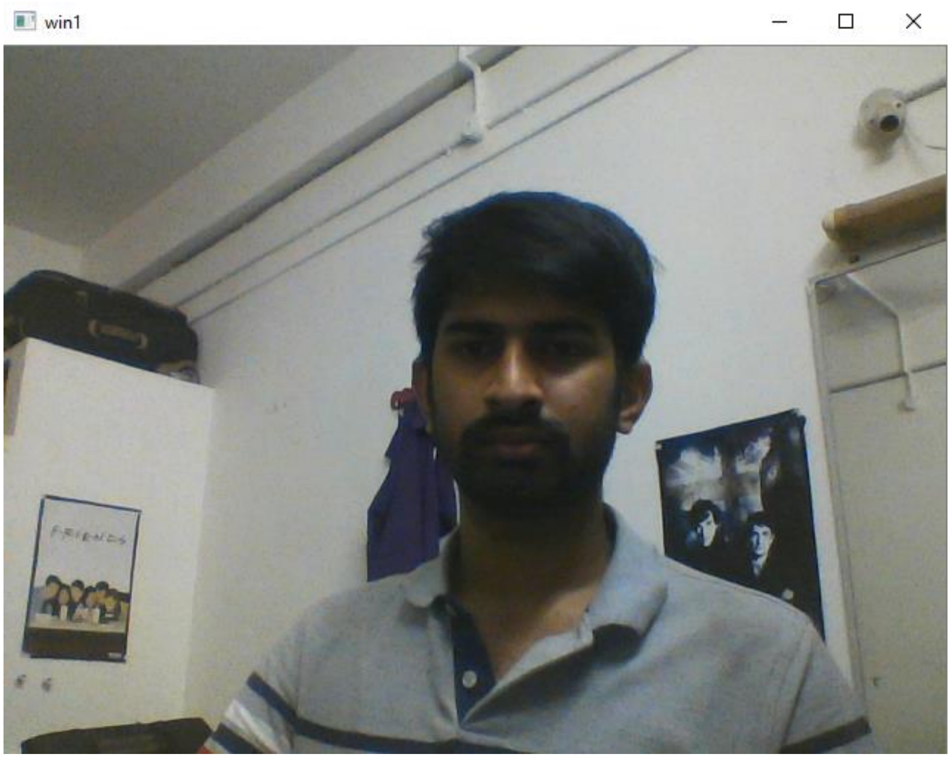


This is the first frame which is captured first. The name of the window is win1.

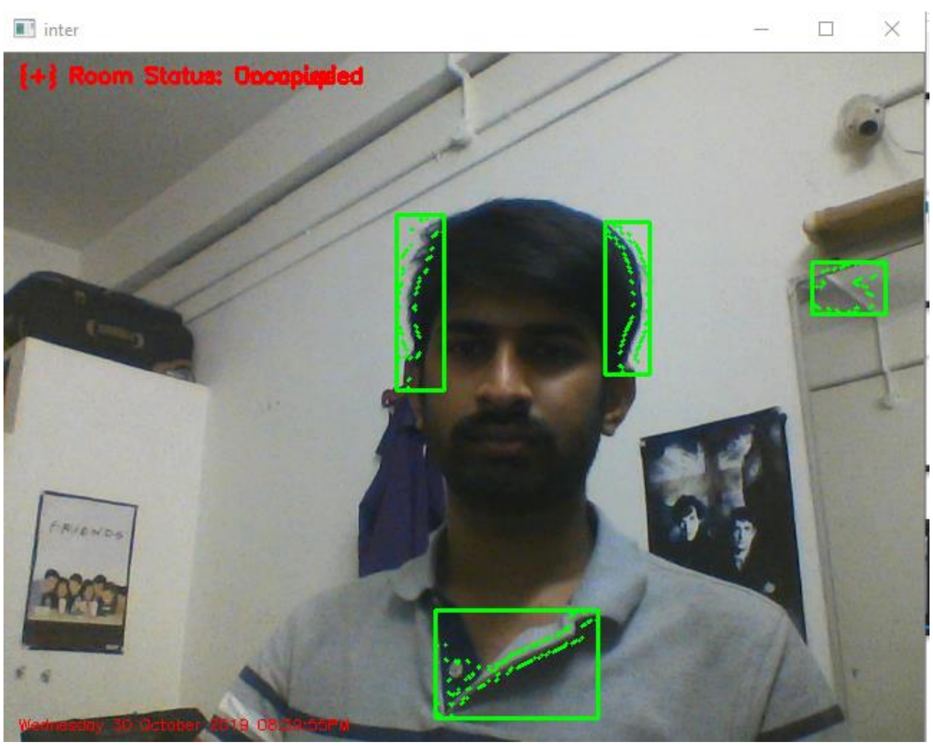


This is the second frame which is captured 2nd after 1ms delay of first frame. The name of window is inter.

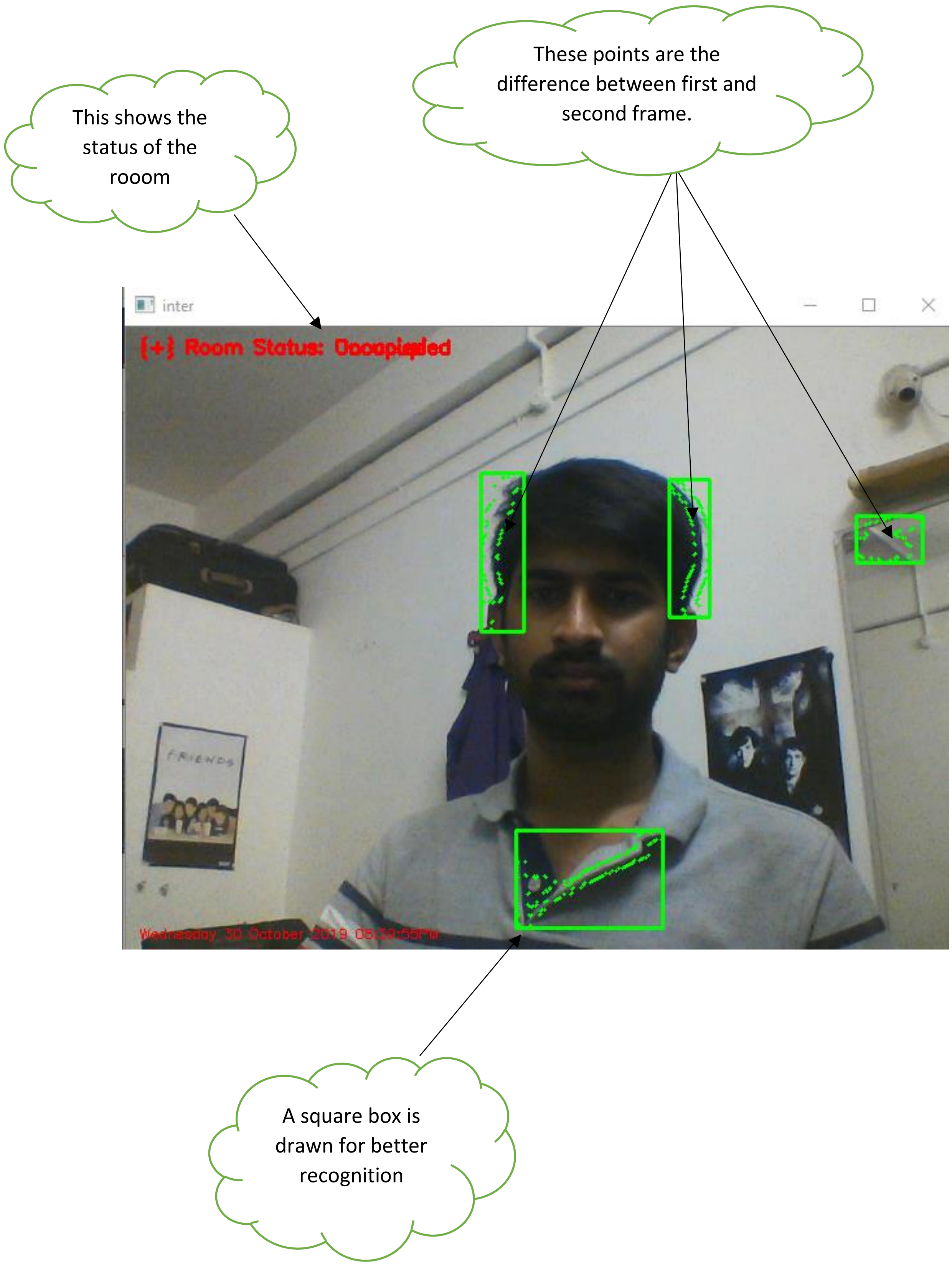
There is no absdiff (absolute differences) between the first and second frame. Hence the status of the room is Unoccupied.



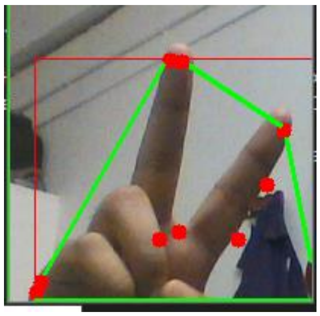
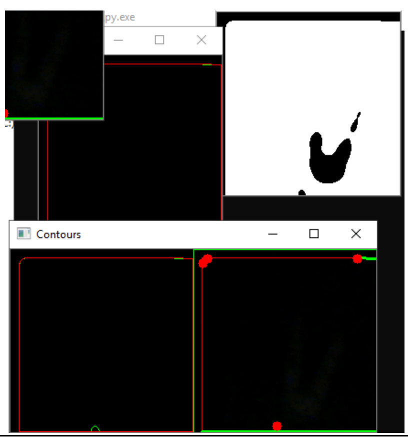
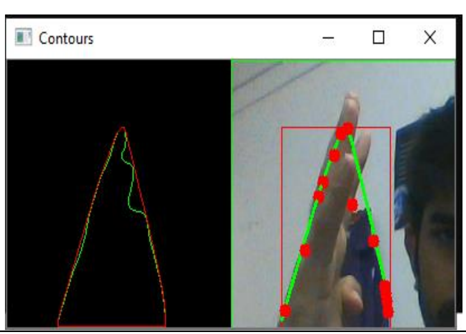
This is the first frame which is captured first. The name of the window is win1.

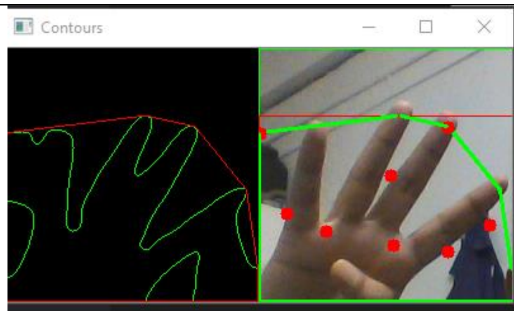
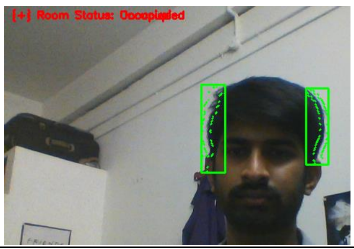
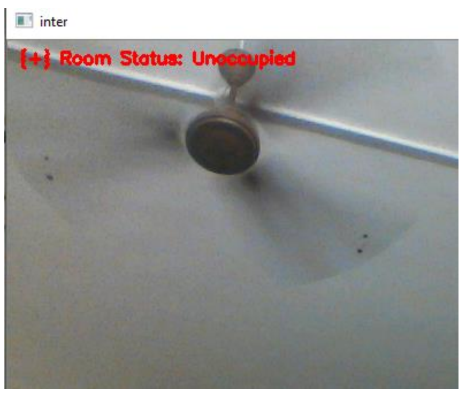


This is the second frame which is captured 2nd after 1ms delay of first frame. The name of window is inter.


This shows the status of the rooom
These points are the difference between first and second frame.
inter
A square box is drawn for better recognition

We check if the no of contours from the absdiff (absolute difference) between the is greater than 1000 or not. If it is, contours are highlighted in neon colour and rectangle boxes are drawn for identifying the movement. So in this case the room status is Occupied.

Test ID	Module Name	Description	Actual Result	Expected Result	Test Case
G01	Gesture Recognition	Here the gesture is pointed when the room is with lights. 	Fan speed 2	Fan speed 2	Pass
G02	Gesture Recognition	Here the gesture is pointed when the room is dark. 	Fan speed 1	Fan speed 2	Fail
G03	Gesture Recognition	Here the fingers are aligned in x-z plane instead of x-y plane 	Fan speed 1	Fan speed 4	Fail
G04	Gesture Recognition	Here the fingers are aligned in x-y plane	Fan speed 4	Fan speed 4	Pass

					
M01	Motion detection	When human is present and motion is detected 	Room Occupied	Room Occupied	Pass
M02	Motion detection	When motion is too fast, it will not get detected.  Here the fan is moving too fast for our program to detect the motion	Room unoccupied	Room Occupied	Fail