Human Interaction with Weaponized Autonomous Surveillance Robots

Group 4
Anita Iotti, Polislava Kalcheva, Martin Karakolev, Jurre Van Opzeeland

Introduction

Our robot is designed to be an autonomous weaponized surveillance system: it is embodied, meaning that its vehicle-like design allows it to move, find and assess targets, and ultimately pursue them by following their movements. More specifically, the robot operates via facial recognition, in order to perform surveillance tasks and to protect itself from potential threats: upon detecting a human face, it activates the trigger mechanism which causes the weapon to shoot (see Appendix A for the related circuit schematics). Additionally, the camera allows the autonomous robot to follow the detected face and to keep the weapon pointing in the appropriate direction: this mechanism is used in order to successfully pursue the target.

The ethical implications of autonomous robotic systems that need to be examined are numerous: for instance, a central theoretical principle concerns the attribution of guilt in situations where a robot causes harm to a human being (Clark, 2022). As the field of social robotics advances, it becomes increasingly important for a developing society to understand the implications of embodied AI systems and autonomous robots that operate with minimal human intervention (Clark, 2022). The nature of human-robot interaction often involves a single activation event, such as a human pressing a button to initiate the robot's operation (Muller & Vincent, 2020): however, as these autonomous systems become more prevalent, concerns arise regarding the potential loss of human control over the robots themselves and the potential for harm to human beings.

Therefore, the increasing autonomy of robots may challenge traditional notions of liability and moral responsibility, necessitating a reevaluation of legal and ethical frameworks to accommodate the growing presence of AI and robotics in everyday life (International Committee of the Red Cross, 2018).

Practical implementation

We employed the following components: 2 DC motors, 4 servo motors, 2 transistors, 1 distance sensor and, lastly, 1 battery pack.

Arduino 1

On this first Arduino board, the distance sensor will check whether an object is present in less than 35cm of range: if nothing is detected, the DC motors will be powered to move forward, otherwise the distance sensor will be angled

towards the right using the servo motor, to then measure the distance again (Arduino - DC Motor, n.d.). Successively, if no object is detected, the left DC motor will be activated first: after a delay, the right DC motor will follow in order to move to the right. In case there is an object detected , the robot will angle the distance sensor towards the left using the servo motor, and then measure the distance again.
Similarly to the previously explained mechanism, if there is no object detected, the right DC motor will be powered first, and again after a delay the left DC motor will do the same to move to the left (Arduino - DC Motor, n.d.). If an object is detected, the right motor will be powered for a longer time, to allow the robot to turn around.
The cycle will then start over.

Arduino 2
Through this second Arduino board, the RGB camera will send a picture of the environment to the computer, which the Python program will try to scan for human faces (Hackster.io, 2021). The program will then send a serial signal to the Arduino with the coordinates of the face: by using these coordinates, the Arduino will send signals to the servo motors in order to move the center of the picture in the direction of the detected face. Once the center of the picture aligns with the detected face, a signal will be sent to the trigger servo motor that activates the actual defense mechanisms by pulling said trigger.

When the robot is left untouched, it will use its motors to move around and search for potential targets. In the meantime, it will be using its distance sensors to make sure it does not drive into any obstacles. The camera system will stay idle until a face is detected in frame.

Ethical considerations

The use of weaponized autonomous surveillance robots raises a number of ethical concerns. One of the most significant ones is the potential for these systems to be hacked and subsequently used for malicious purposes (Clark, 2022): it would then be difficult to determine who is responsible for their actions. Clark (2022), in fact, poses the following question: if a robot kills someone, who is responsible for that death? Is it the robot's manufacturer, the person who deployed the robot, or the robot itself?
Another relevant issue to tackle is the inevitable change of the underlying nature of wars: as embodied AI systems and autonomous robots become progressively more efficient, and therefore less dependent on human control, the central mechanisms of these conflicts must be completely revisited and renewed (Clark, 2022).

Additional concerns address the use of input and information needed for these autonomous robots in order to guide their behavior (Muller & Vincent, 2020): due to their very nature, autonomous weapon systems can select and engage targets without further intervention by a human operator once activated, as stated by the United States' Department of Defense. It is therefore argued that such machines might promote additional deaths by delegating responsibility from humans (Muller & Vincent, 2020). Muller & Vincent (2020) also pose a number of potential additional matters that might insurge as better autonomous weaponized machines are developed, such as lack of human control and accountability (as previously highlighted by the aforementioned paper), potential increased risk of accidents, as well as increased risk of cyber attacks.

It is important to add that autonomous weapon systems have also been suggested to eventually result in a more efficient choice after results-oriented analysis: their potential precision and reliability might enable better respect for both international law and human ethical values, resulting in fewer adverse humanitarian consequences (International Committee of the Red Cross, 2018).

To summarize, significant concerns address the lack of human control over these robots and their ability to make decisions that could result in harm (International Committee of the Red Cross, 2018; Clark, 2022; Muller & Vincent, 2020): although autonomous weaponized robots might be the most efficient choice under certain aspects, the legal and ethical issues raised by these systems require thorough analysis and will need to be addressed in the near future, in order to create a safer environment and common guidelines.

Discussion

The robot was presented and tested during demo day: when activated, the robot tried to find its target using facial recognition and employed the toy gun (when clocked) to fire at said target.

The agent behaved as expected, although some work on its mobility is still required. The performance of the facial recognition software was consistent, without any major mistakes. In this current version of the robot the balance is not very good, as there were only two big and one small wheel attached for support: this was a mistake on our side as we failed to utilize more than one Arduino kit. Furthermore, the construction of the arm which holds the servo motor, the camera and the toy gun was not very well engineered which led to a problem of "shaking" at some stage, and therefore the robot was not able to focus on a target for more than a few seconds. We managed to work around this problem by placing some small plastic bricks to help with the arm support.

For future prototypes we think it will be useful for the robot to have 4 big wheels instead of only 2 as that would increase its mobility, as our current prototype is not very mobile. For the armed part, a bigger radius of rotation is needed: that way more ground can be covered without the need for the agent to change positions or needing outside assistance.

The use of H bridges instead of the transistors we received with the Arduino kit will be a very useful addition to the robot as they are more efficient, better at braking the motor and have a better rotation direction change. Moreover, the robot's facial recognition can be improved, if needed, to target a specific face instead of just any random face that appears.

Other uses for the robot include the removal of the toy gun and its replacement with a charging cable for EVs, making the facial recognition target ports instead of faces: that way, this system can be used to charge people's cars instead of potentially causing harm.

Overall, our robot performs repeatedly well on the tasks it was designed for. It appears to be autonomous, as it does not need any human input to select a target and attempt to shoot. The interaction with humans was very positive given the nature of the robot: users seemed impressed with the consistency of the facial recognition and the accuracy with which our agent was able to perform the task. If a new prototype is to be build, then improvement of the mobility and the stability of the robot has to

be made using the aforementioned possible upgrades.

References:

Arduino - DC Motor. (n.d.). https://www.tutorialspoint.com/arduino/arduino_dc_motor.htm

Clark, S. (2022). Part of the kill chain: how can we control weaponised robots? *The Guardian*. https://www.theguardian.com/technology/2022/nov/20/part-of-the-kill-chain-how-can-we-control-weaponised-robots

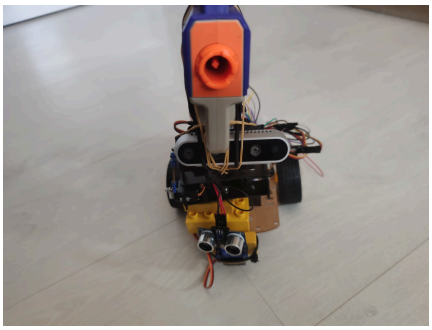
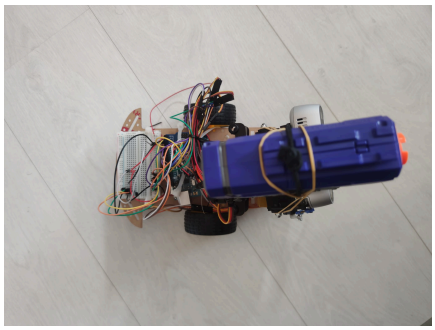
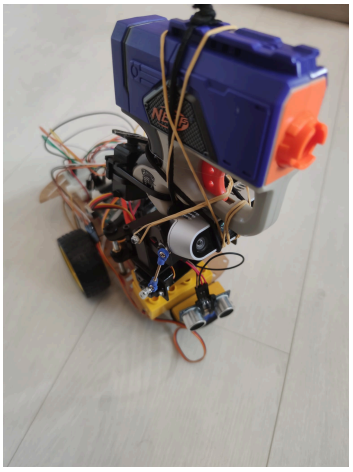
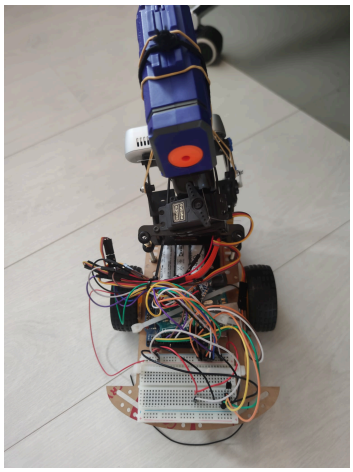
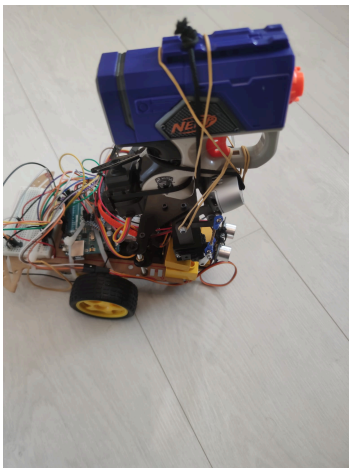
Hackster.io. (2021, February 2). *Face Tracker Using OpenCV and Arduino*. https://www.hackster.io/shubhamsantosh99/face-tracker-using-opencv-and-arduino-55412e

International Committee of the Red Cross (2018). *Ethics and autonomous weapon systems: An ethical basis for human control?* https://www.icrc.org/en/document/ethics-and-autonomous-weapon-systems-ethical-basis-human-control

Müller, V. C. (2020). Ethics of artificial intelligence and robotics. *Stanford Encyclopedia of Philosophy*. https://philpapers.org/archive/MLLEOA-4.pdf

Appendix A

Figures 1-5: An overview of what the finished agent looks like

Appendix B

Arduino 1 code:

```
#include<Servo.h>
const int trigPin = 6;
const int echoPin = 7;
int motorPinL = 3;
int motorPinR = 2;
long duration;
int distance;
Servo s;

int calcdistance(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

```
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.println(distance);
  return(distance);
}

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  s.attach(8);
  s.write(90);
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  if (calcdistance() >= 35) {
    analogWrite(motorPinL, 255);
    analogWrite(motorPinR, 255);
    digitalWrite(LED_BUILTIN, HIGH);
    Serial.print("Forwards");
  }
  else {
    s.write(0);
    delay(1000);
    if (calcdistance() >= 35) {
    analogWrite(motorPinL, 255);
    delay(500);
    analogWrite(motorPinL, 255);
    analogWrite(motorPinR, 255);
    digitalWrite(LED_BUILTIN, HIGH);
    Serial.print("Right");
    }
    else {
      s.write(180);
      delay(1000);
      if (calcdistance() >= 35) {
      analogWrite(motorPinR, 255);
      delay(500);
      analogWrite(motorPinL, 255);
```

```
      analogWrite(motorPinR, 255);
      digitalWrite(LED_BUILTIN, HIGH);
      Serial.print("Left");
      }
      else {
      analogWrite(motorPinR, 255);
      digitalWrite(LED_BUILTIN, HIGH);
      Serial.print("Backwards");
      delay(1000);
      }
    }
  }
  delay(500);
  analogWrite(motorPinL, 0);
  analogWrite(motorPinR, 0);
  s.write(90);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);

}
```

Arduino 2 code:

```
#include<Servo.h>

Servo x, y, g;
int width = 640, height = 480;
int xpos = 100, ypos = 60, gpos = 180;
void setup() {

  Serial.begin(9600);
  x.attach(9);
  y.attach(10);
  g.attach(11);
  x.write(xpos);
  y.write(ypos);
  g.write(gpos);
}
const int angle = 1;

void loop() {
```

```
if (Serial.available() > 0)
{
  int x_mid, y_mid;
  if (Serial.read() == 'X')
  {
    x_mid = Serial.parseInt();
    if (Serial.read() == 'Y')
      y_mid = Serial.parseInt();
  }
  if (x_mid > width / 2 + 30)
    xpos -= angle;
  if (x_mid < width / 2 - 30)
    xpos += angle;
  if (y_mid < height / 2 + 30)
    ypos += angle;
  if (y_mid > height / 2 - 30)
    ypos -= angle;

  if ((x_mid < width / 2 + 30) && (x_mid > width / 2 - 30))
    {
      if ((y_mid < height / 2 + 30) && (y_mid > height / 2 - 30))
        gpos = 0;
    }

  if (xpos >= 160){
    xpos = 160;
  }
  else if (xpos <= 60){
    xpos = 60;
  }
  if (ypos >= 110){
    ypos = 110;
  }
  else if (ypos <= 0){
    ypos = 0;
  }

  x.write(xpos);
  y.write(ypos);
  g.write(gpos);
```

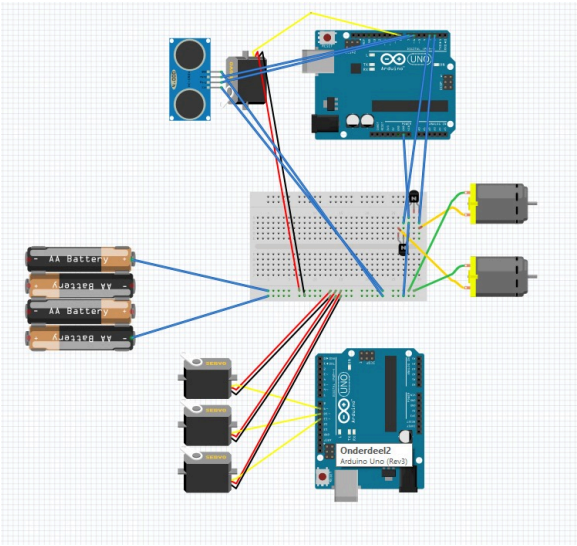
```
    gpos = 180;
  }
```

Python code:

```
import cv2
import serial,time
face_cascade=
cv2.CascadeClassifier('haarcascade_frontalface_default
.xml') 2
cap=cv2.VideoCapture(2)
ArduinoSerial=serial.Serial('com3',9600,timeout=0.1)
time.sleep(1)

while cap.isOpened():
   ret, frame= cap.read()
   frame=cv2.flip(frame,1)
   gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
   faces= face_cascade.detectMultiScale(gray,1.1,6)
   for x,y,w,h in faces:
       string='X{0:d}Y{1:d}'.format((x+w//2),(y+h//2))
       print(string)
       ArduinoSerial.write(string.encode('utf-8'))
       cv2.circle(frame,(x+w//2,y+h//2),2,(0,255,0),2)

cv2.rectangle(frame,(x,y),(x+w,y+h),(0,0,255),3)
   cv2.rectangle(frame,(640//2-30,480//2-30),
                 (640//2+30,480//2+30),
                  (255,255,255),3)
   cv2.imshow('img',frame)
   if cv2.waitKey(10)&0xFF== ord('q'):
       break
cap.release()
cv2.destroyAllWindows()
```

Appendix C

Circuit schematics for therobot’s mechanisms.