APPENDICES

1. Specifications of the Used Components

Table 1: Specifications of the robotic (claw) arm

Parameters	Values
No. of Servo Mounted	4
Arm Base Orientation	180°
Weight Capacity	200 gm
Max Forward Length from Base	17 cm
Max Height from Base	16 cm
Claw Length (Fit)	2 cm

Table 2: Specifications of the servo motors

Parameters	Values
Operating Voltage	+5V
Torque	2.5 kg/cm
Operating Speed	0.1 s/60°
Gear Type	Plastic
Rotation	0° - 180°
Weight of Motor	9 gm

Table 3: Specifications of 12V 5A AC Power Adapter

Parameters	Values
Input Voltage	50/60 Hz 100 - 240V
Output Voltage	12 V DC
Output Current	5A
Output Interface	5.5 x 2.5 mm
Power	60W
Temperature	0 ~ +50℃

Table 4: Specifications of 12V DC Motor

Parameters	Values
Voltage	12V DC
Gear Ratio	1:75
Speed (RPM)	50 +/- 10% (Without Load) 40 +/- 13% (With Load)
Current (mA)	= < 100 (Without Load) = < 250 (With Load)
Max Weight	142 gm
Max size (Diameter x Length)	24.4 mm x 65.8 mm

Table 5: Specifications of the Arduino Uno Microcontroller

Parameters	Values
Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage	6 - 20V
No. of Digital I/O Pins	14
No. of PWM Digital I/O Pins	6
No. Analog Input Pins	6
DC Current for 3.3V Pin	50 mA
DC Current per I/O Pin	20 mA
Clock Speed	16 MHz
Dimensions (Length x Width)	68.6 mm x 58.4 x mm
Weight	25 g

*Corresponding Author Email Address: u1603050@student.cuet.ac.bd

CODE FOR THE SYSTEM

```
import cv2

cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 360)

while True:
    _, frame = cap.read()
    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    height, width, _ = frame.shape

    cx = int(width / 2)
    cy = int(height / 2)

    # pick pixel value
    pixel_center = hsv_frame[cy, cx]
    hue_value = pixel_center[0]

    color = "Undefined"
    if 0 < hue_value < 7:
        color = "RED"

        # import Real_Time_Shape_Red
        import cv2
        import numpy as np

        def nothing(x):
            pass

        cap = cv2.VideoCapture(0)

        cv2.namedWindow("Trackbars")
        cv2.resizeWindow("Trackbars", 500, 500)
        cv2.createTrackbar("L-H", "Trackbars", 0, 180, nothing)
        cv2.createTrackbar("L-S", "Trackbars", 160, 255, nothing)
        cv2.createTrackbar("L-V", "Trackbars", 160, 255, nothing)
        cv2.createTrackbar("U-H", "Trackbars", 180, 180, nothing)
        cv2.createTrackbar("U-S", "Trackbars", 255, 255, nothing)
        cv2.createTrackbar("U-V", "Trackbars", 255, 255, nothing)

        font = cv2.FONT_HERSHEY_COMPLEX

        while True:
            _, frame = cap.read()
            hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

            l_h = cv2.getTrackbarPos("L-H", "Trackbars")
            l_s = cv2.getTrackbarPos("L-S", "Trackbars")
            l_v = cv2.getTrackbarPos("L-V", "Trackbars")
            u_h = cv2.getTrackbarPos("U-H", "Trackbars")
            u_s = cv2.getTrackbarPos("U-S", "Trackbars")
            u_v = cv2.getTrackbarPos("U-V", "Trackbars")

            lower_red = np.array([l_h, l_s, l_v])
            upper_red = np.array([u_h, u_s, u_v])

            mask = cv2.inRange(hsv, lower_red, upper_red)
            kernel = np.ones((5, 5), np.uint8)
            mask = cv2.erode(mask, kernel)
```

```
# Contours Detection
contours, _ = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

for cnt in contours:
    area = cv2.contourArea(cnt)
    approx = cv2.approxPolyDP(cnt, 0.01 * cv2.arcLength(cnt, True), True)
    x = approx.ravel()[0]
    y = approx.ravel()[1]

    if area > 400:
        cv2.drawContours(frame, [approx], 0, (0, 0, 0), 5)

        if len(approx) <= 6:
            cv2.putText(frame, "Red Rectangle", (x, y), font, 1, (0, 0, 0))

            #import Arm_3

            import pyfirmata
            import time

            board = pyfirmata.Arduino('COM5')
            base = board.get_pin('d:5:s')
            left = board.get_pin('d:7:s')
            right = board.get_pin('d:9:s')
            claw = board.get_pin('d:11:s')

            base.write(90)
            time.sleep(2)
            right.write(60)
            time.sleep(2)
            left.write(160)
            time.sleep(2)
            claw.write(90)
            time.sleep(2)
            for angle in range(60, 90, 1):
                right.write(angle)
                time.sleep(0.015)
            claw.write(0)
            time.sleep(2)
            right.write(60)
            time.sleep(2)
            base.write(200)
            time.sleep(2)
            claw.write(90)
            time.sleep(2)
            claw.write(0)
            time.sleep(2)
            base.write(90)
            time.sleep(2)

            cap.release()
            cv2.destroyAllWindows()

        elif len(approx) >= 6:
            cv2.putText(frame, "Red Circle", (x, y), font, 1, (0, 0, 0))

            #import Arm_4

            import pyfirmata
            import time

            board = pyfirmata.Arduino('COM5')
```

```
                base = board.get_pin('d:5:s')
                left = board.get_pin('d:7:s')
                right = board.get_pin('d:9:s')
                claw = board.get_pin('d:11:s')

                base.write(90)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                left.write(160)
                time.sleep(2)
                claw.write(90)
                time.sleep(2)
                for angle in range(60, 100, 1):
                    right.write(angle)
                    time.sleep(0.015)
                claw.write(0)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                base.write(135)
                time.sleep(2)
                claw.write(90)
                time.sleep(2)
                claw.write(0)
                time.sleep(2)
                base.write(90)
                time.sleep(2)

                cap.release()
                cv2.destroyAllWindows()

            cv2.imshow("Frame", frame)
            cv2.imshow("Mask", mask)

            key = cv2.waitKey(1)
            if key == 27:
                break

        cap.release()
        cv2.destroyAllWindows()

    elif 99 < hue_value < 105:
        color = "BLUE"

        # import Real_Time_Shape_Blue
        import cv2
        import numpy as np

        def nothing(x):
            pass

        cap = cv2.VideoCapture(0)

        cv2.namedWindow("Trackbars")
        cv2.resizeWindow("Trackbars", 500, 500)
        cv2.createTrackbar("L-H", "Trackbars", 70, 180, nothing)
        cv2.createTrackbar("L-S", "Trackbars", 70, 255, nothing)
        cv2.createTrackbar("L-V", "Trackbars", 180, 255, nothing)
        cv2.createTrackbar("U-H", "Trackbars", 180, 180, nothing)
        cv2.createTrackbar("U-S", "Trackbars", 255, 255, nothing)
        cv2.createTrackbar("U-V", "Trackbars", 255, 255, nothing)
```

```
font = cv2.FONT_HERSHEY_COMPLEX

while True:
    _, frame = cap.read()
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    l_h = cv2.getTrackbarPos("L-H", "Trackbars")
    l_s = cv2.getTrackbarPos("L-S", "Trackbars")
    l_v = cv2.getTrackbarPos("L-V", "Trackbars")
    u_h = cv2.getTrackbarPos("U-H", "Trackbars")
    u_s = cv2.getTrackbarPos("U-S", "Trackbars")
    u_v = cv2.getTrackbarPos("U-V", "Trackbars")

    lower_red = np.array([l_h, l_s, l_v])
    upper_red = np.array([u_h, u_s, u_v])

    mask = cv2.inRange(hsv, lower_red, upper_red)
    kernel = np.ones((5, 5), np.uint8)
    mask = cv2.erode(mask, kernel)

    # Contours Detection
contours, _ = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    for cnt in contours:
        area = cv2.contourArea(cnt)
        approx = cv2.approxPolyDP(cnt, 0.01 * cv2.arcLength(cnt, True), True)
        x = approx.ravel()[0]
        y = approx.ravel()[1]

        if area > 400:
            cv2.drawContours(frame, [approx], 0, (0, 0, 0), 5)

            if len(approx) <= 7:
                cv2.putText(frame, "Blue Rectangle", (x, y), font, 1, (0, 0, 0))

                #import Arm_1

                import pyfirmata
                import time

                board = pyfirmata.Arduino('COM5')
                base = board.get_pin('d:5:s')
                left = board.get_pin('d:7:s')
                right = board.get_pin('d:9:s')
                claw = board.get_pin('d:11:s')

                base.write(90)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                left.write(160)
                time.sleep(2)
                claw.write(90)
                time.sleep(2)
                for angle in range(60, 90, 1):
                    right.write(angle)
                    time.sleep(0.015)
                claw.write(0)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                base.write(0)
                time.sleep(2)
```

```
                claw.write(90)
                time.sleep(2)
                claw.write(0)
                time.sleep(2)
                base.write(90)
                time.sleep(2)

                cap.release()
                cv2.destroyAllWindows()

            elif len(approx) >= 7:
                cv2.putText(frame, "Blue Circle", (x, y), font, 1, (0, 0, 0))

                #import Arm_2

                import pyfirmata
                import time

                board = pyfirmata.Arduino('COM5')
                base = board.get_pin('d:5:s')
                left = board.get_pin('d:7:s')
                right = board.get_pin('d:9:s')
                claw = board.get_pin('d:11:s')

                base.write(90)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                left.write(160)
                time.sleep(2)
                claw.write(90)
                time.sleep(2)
                for angle in range(60, 100, 1):
                    right.write(angle)
                    time.sleep(0.015)
                claw.write(0)
                time.sleep(2)
                right.write(60)
                time.sleep(2)
                base.write(45)
                time.sleep(2)
                claw.write(90)
                time.sleep(2)
                claw.write(0)
                time.sleep(2)
                base.write(90)
                time.sleep(2)

                cap.release()
                cv2.destroyAllWindows()

        cv2.imshow("Frame", frame)
        cv2.imshow("Mask", mask)

        key = cv2.waitKey(1)
        if key == 27:
            break

    cap.release()
    cv2.destroyAllWindows()

pixel_center_bgr = frame[cx, cy]
cv2.putText(frame, color, (10, 50), 0, 1, (255, 0, 0), 2)
```

```
    cv2.circle(frame, (cx, cy), 5, (255, 0, 0), 3)

    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1)
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```