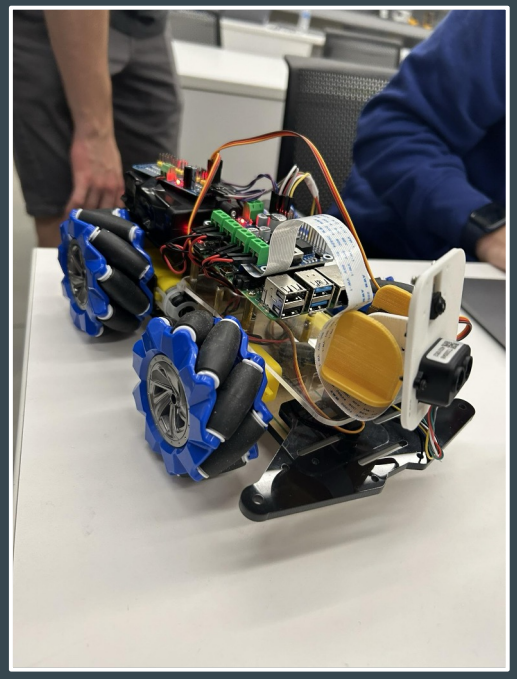
Team 3

Assembled by Cole, Zaid, Zesen, and Davoud Ghassemiyeh

Table of Contents



Introduction and Purpose

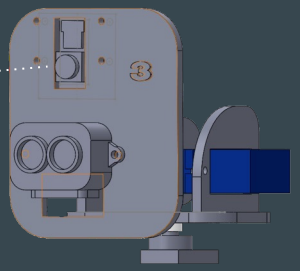
- The rover is comprised of a 4-wheel drive kit with omnidirectional wheels. Included in the kit were the wheels, four motors, and a handful of plastic platforms to elevate and separate our PCB boards.
- In addition to this kit, we added our own custom mounting brackets that will allow the rover to utilize servos to pan and tilt both a camera and a TF Luna (both given by professor).
- The main purpose of this rover is to detect and follow threats while actively avoiding obstacles.
- We want our rover to survey an area to detect for possible targets (in this case human faces).
- The rover will utilize its pan and tilt features to find and follow threats.
- If there are visible threats, then the rover will follow the target at a safe distance.

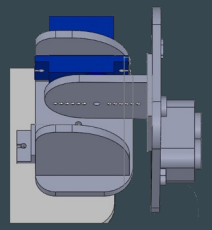
Zaid

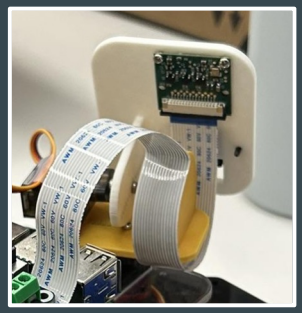
Components Used

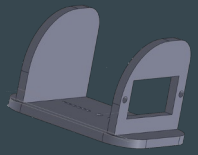
New Components:

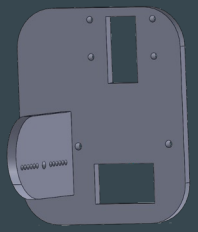
1) Camera
2) TF Luna Sensor
3) Custom Pan mounting bracket
4) Custom Tilt mounting bracket

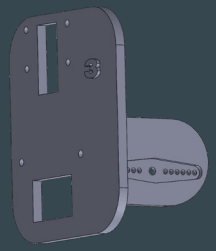


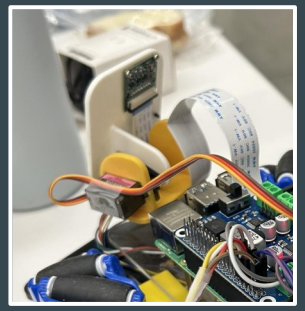


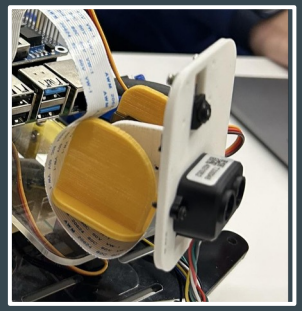








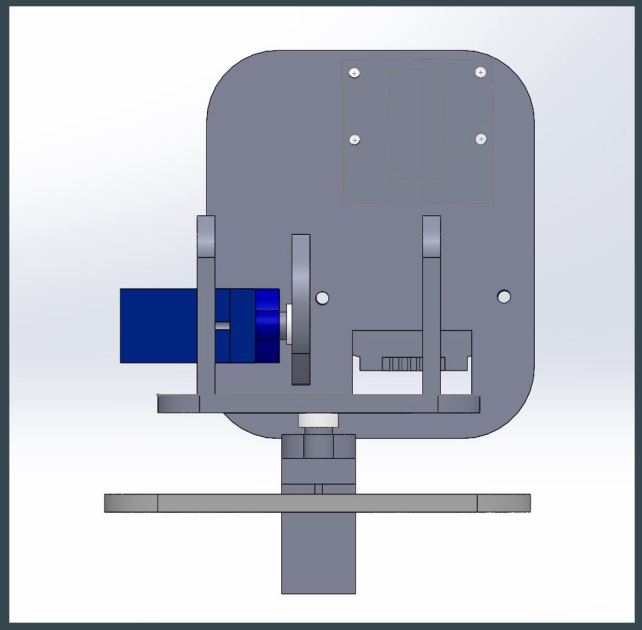


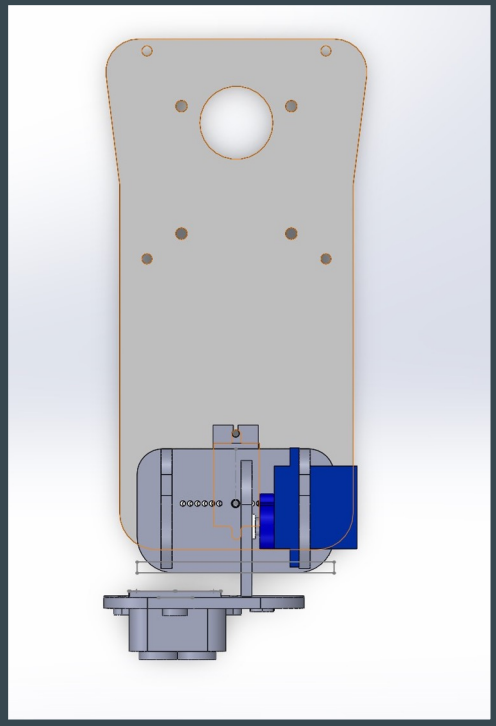


Zaid

SolidWorks Design

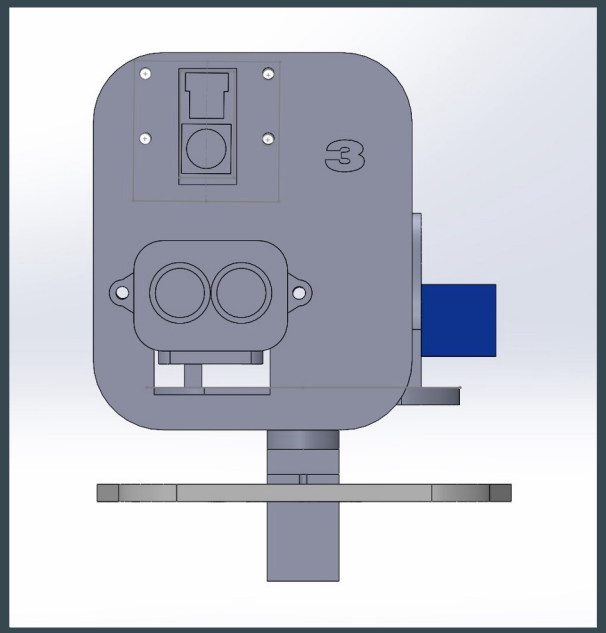
- In order to achieve success, we needed to create a bracket that housed both pan and tilt features.
- As we made countless revisions in our design, we reached a point that was crucial to the rover's performance.
- Our final design specified locations for each part which was a key to our rover operating flawlessly.

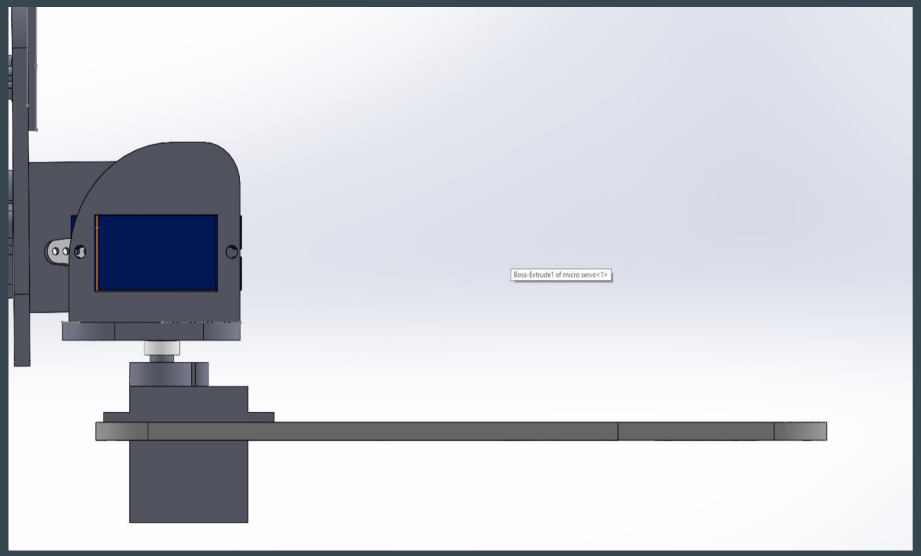




Davoud

Cont.



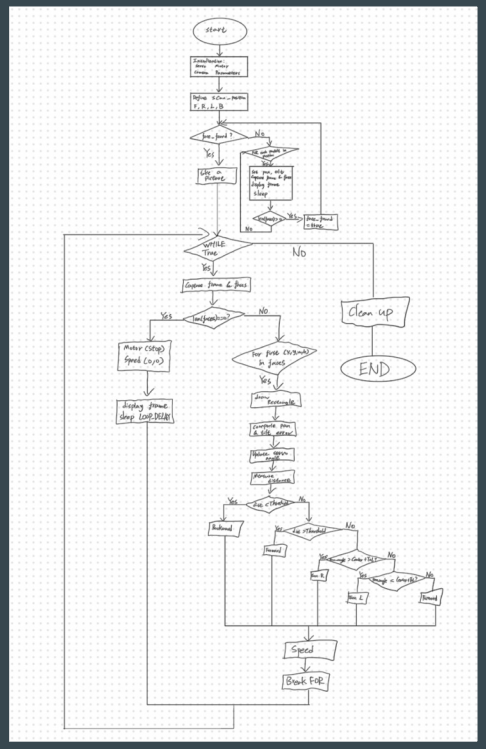


Key Features

- By adding new hardware and redesigning key parts, we were able to add new features to our rover.
- The camera gives the rover ability to do Face & Eye recognition.
- The Pan function gives the rover ability to move the camera 180° on the X, Y plane which allows it to detect and collect data from left to right.
- The Tilt function improves mobility on the Z, X plane around 45°, which gives us the ability to look up and down.

Davoud

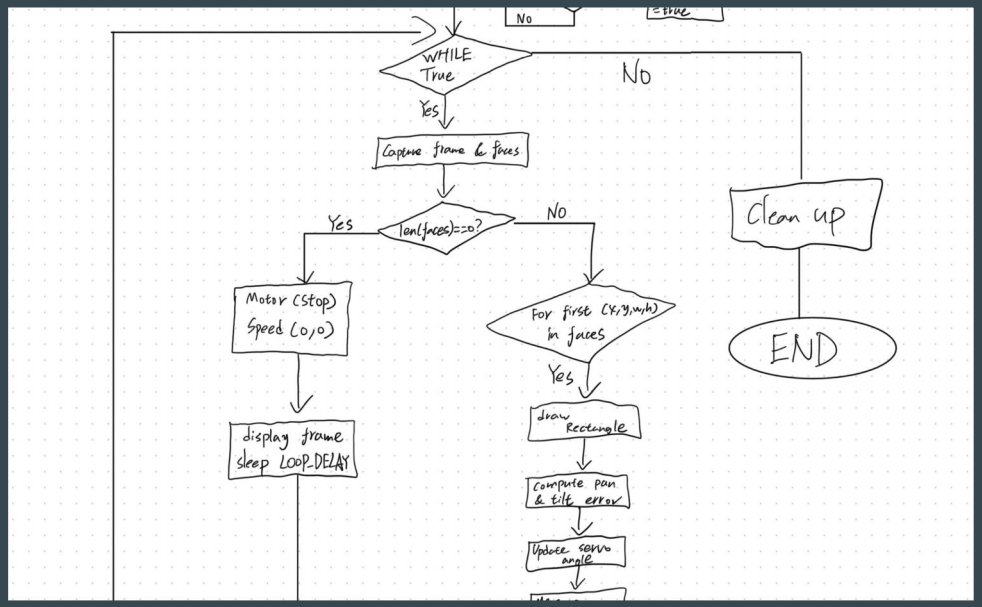
Flowchart & Sample Code




```python
# --- 1) Initial 5?Way Scan ---
scan_positions = [
    (90,  90),   # center
    (30,  90),   # left
    (150, 90),   # right
    (90,  45),   # up
    (90,  150)   # down
]

face_found = False
while not face_found:
    for pan, tilt in scan_positions:
        Team3_Rover.set_pan_tilt(pan, tilt)
        frame, faces = Team3_Rover.capture_frame_and_faces()
        cv2.imshow("Camera", frame)
        cv2.waitKey(1)
        time.sleep(SCAN_PAUSE)
        if len(faces) > 0:
            face_found = True
            break

# --- 2) Take one snapshot ---
Team3_Rover.take_snapshot()
```

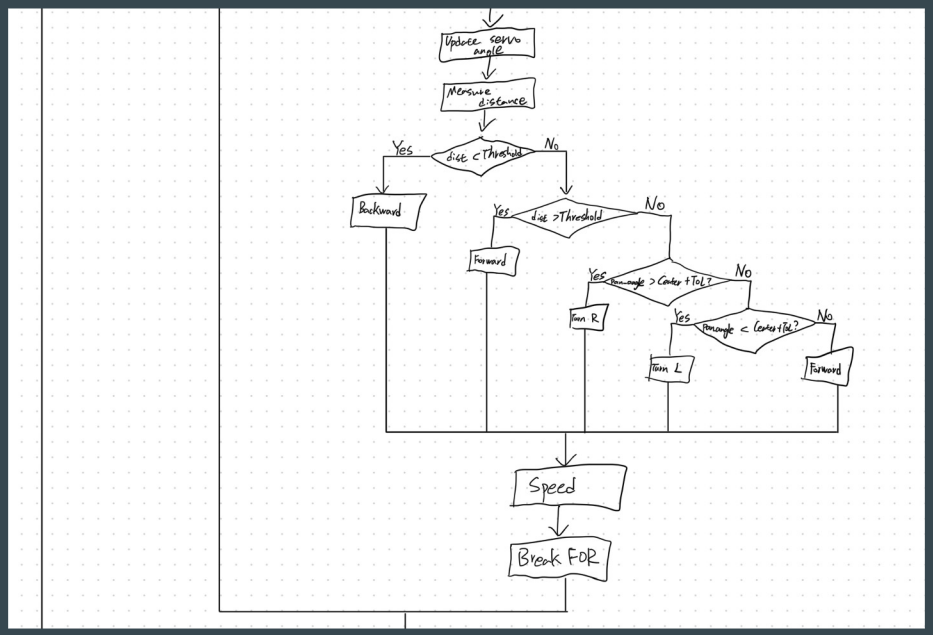
Zesen

Flowchart & Sample Code (Continued)




```python
# --- 3) Continuous Tracking Loop ---
try:
    while True:
        frame, faces = Team3_Rover.capture_frame_and_faces()

        # 3a) No face ? STOP
        if len(faces) == 0:
            Team3_Rover.Motor(0,0,0,0,0,0,0,0)
            Team3_Rover.Speed(BASE_SPEED, BASE_SPEED)
            cv2.imshow("Camera", frame)
            cv2.waitKey(1)
            time.sleep(LOOP_DELAY)
            continue

        # 3b) Process only the first detected face
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (255,0,0), 3)

            # Compute pixel errors
            pan_error  = (x + w/2) - (Team3_Rover.DISP_W  / 2)
            tilt_error = (y + h/2) - (Team3_Rover.DISP_H  / 2)

            # Center the face with servos
            pan_angle  = max(45, min(150, pan_angle  - pan_error  / SERVO_SCALE))
            tilt_angle = max(45, min(150, tilt_angle - tilt_error / SERVO_SCALE))
            Team3_Rover.set_pan_tilt(pan_angle, tilt_angle)

            # Measure forward distance
            dist = Team3_Rover.MeasureDistance()
```

Flowchart & Sample Code (Continued)




```
        # 3c) Decision & Actuation
        if dist < DIST_THRESHOLD:
            # Too close ? BACKWARD
            Team3_Rover.Motor(1,0,1,0,1,0,1,0)
        elif dist > DIST_THRESHOLD:
            # Face moved away ? FORWARD
            Team3_Rover.Motor(0,1,0,1,0,1,0,1)
        else:
            # Within threshold ? steer by pan_angle
            if pan_angle > CENTER_ANGLE + ANGLE_TOLERANCE:
                # servo pointed right ? TURN RIGHT
                Team3_Rover.Motor(1,0,0,1,1,0,0,1)
            elif pan_angle < CENTER_ANGLE - ANGLE_TOLERANCE:
                # servo pointed left ? TURN LEFT
                Team3_Rover.Motor(0,1,1,0,0,1,1,0)
            else:
                # servo centered ? FORWARD
                Team3_Rover.Motor(0,1,0,1,0,1,0,1)
```

Challenges

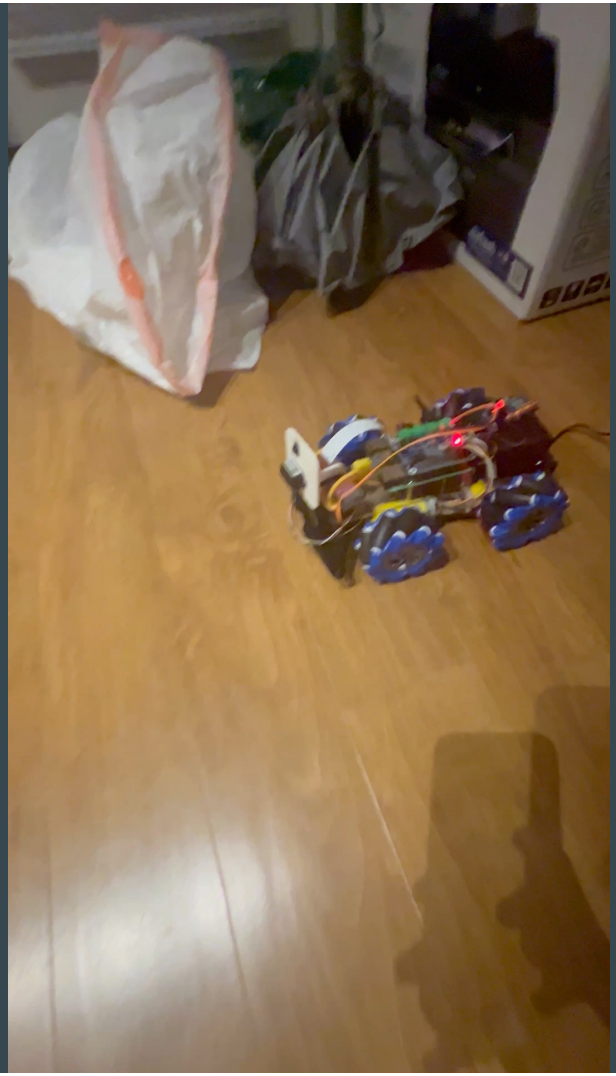
- Problems with our newly printed parts
- Weak material of our parts
- Confusion in our coding for our pan and tilt to work together simultaneously
- Scanning faces and ability to stay in contact and follow with the face
- Determining the correct code setup to achieve our goal by making tiny adjustments.
- Camera reacting sluggish when presented with an intruder.

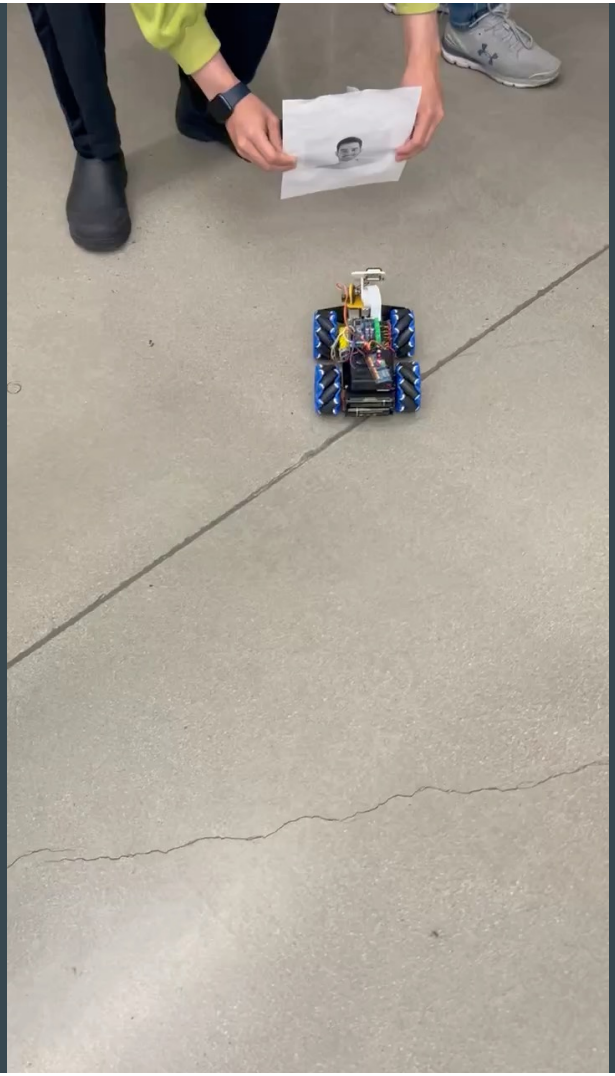
Cole

Conclusion

- All in all, as a team we learned to be patient with our rover and realize we would encounter many errors with our rover's functionality.
- Many things out of our control occurred, but we stood persistent and refined the things most important to our rover.
- As our timeline shortened, we realized many things needed work... and fast!
- After many trials and tirelessly inputting code to maneuver our rover to act as security, a peak point was reached and we ruled this to be its final form.
- Next time, a camera with better abilities could be budgeted better and/or accompanied with updated software and parts to improve visibility and clarity of our subjects.

Cole





Thanks for listening