Team 10:
Matthew Haahr (MH)
Brian Shin (BS)
Nick Hom (NH)

RBE 2002: Unified Robotics II—Lab 4: Position Tracking Post- Lab

Contribution Statement: For this lab, work was divided well. Matt took on the bulk of the programming and testing of the robot. Brian helped debug and worked on the testing. Everyone worked on the documentation of this lab and answered the questions.

1. **Task 1**
    a. The speed used for driving straight was 50 mm/s for both wheels.
    b. Plot

Figure 1:

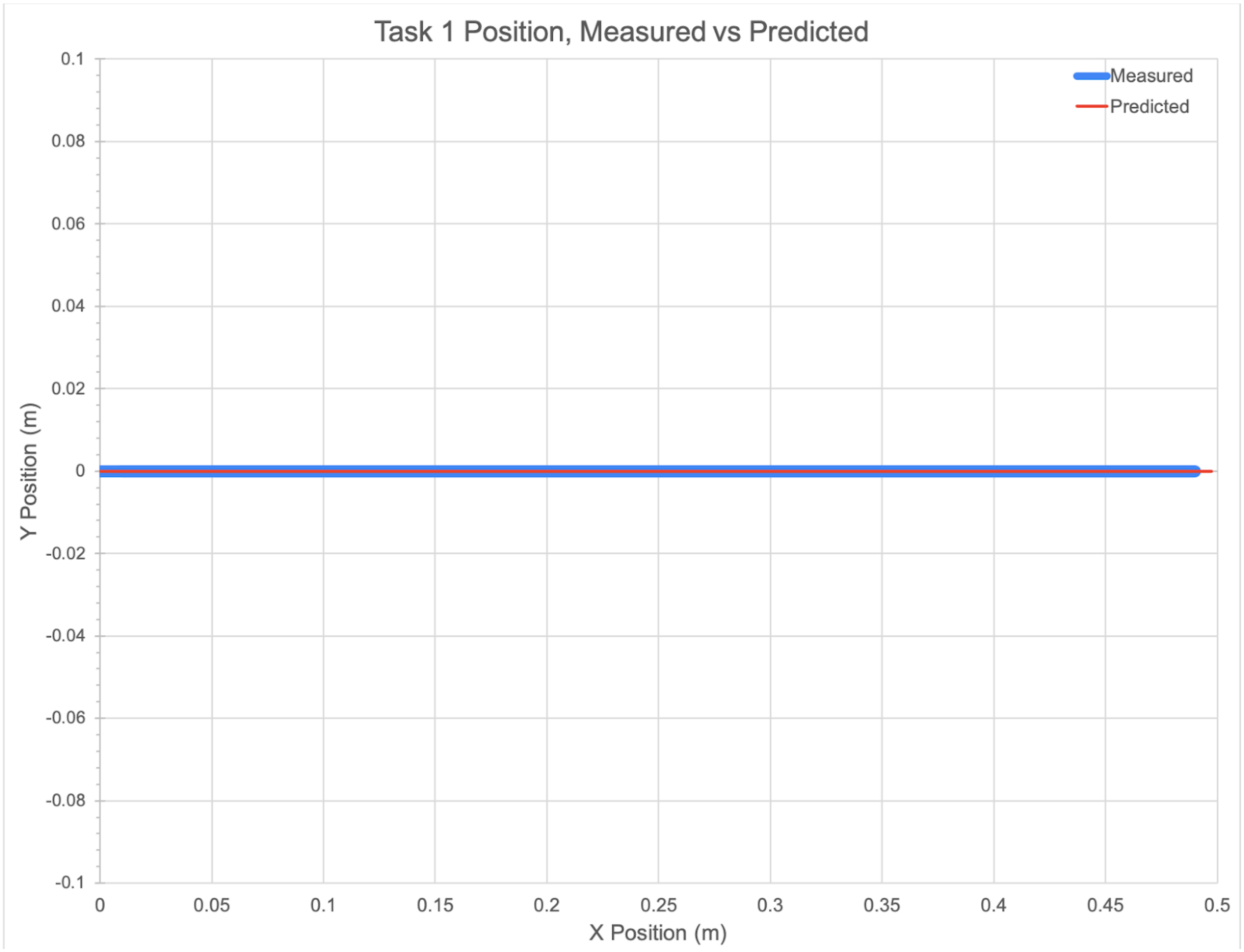

Figure 1: x and y position, both measured and predicted of the Romi driving straight for 10 seconds, turning 180˚, and then driving straight for 10 seconds.

2. **Task 2**

Team 10:
Matthew Haahr (MH)
Brian Shin (BS)
Nick Hom (NH)

a. The speed used for driving straight was 50 mm/s for both wheels and for curved driving, the left wheel speed was 25 mm/s and the right wheel speed was 75 mm/s.
b. Plot

Figure 2:

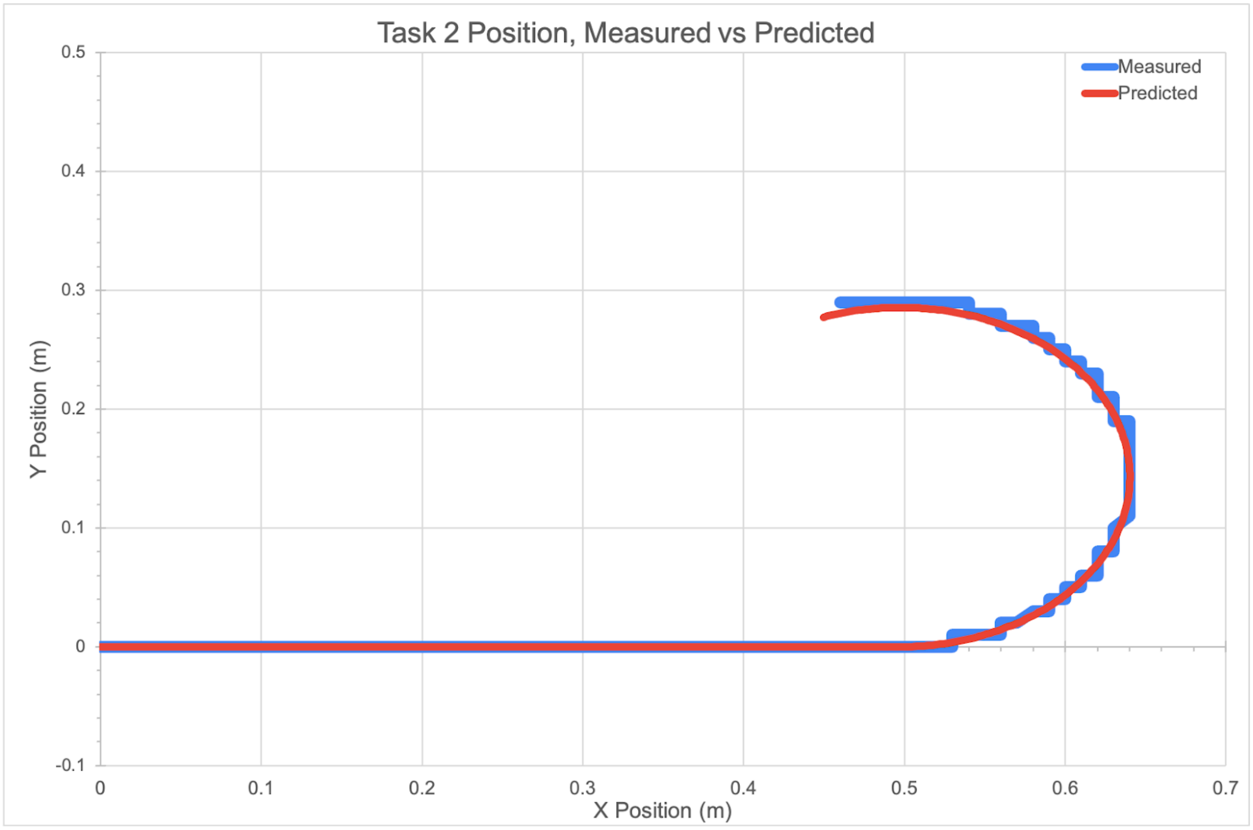

Figure 2: x and y position, both measured and predicted of the Romi driving straight for 10 seconds then driving a curved path for 10 seconds.

3. **Task 3**
   a. The left wheel speed was 120 mm/s while the right wheel speed was 150 mm/s for driving this path
   b. Plot

Figure 3:

Team 10:
Matthew Haahr (MH)
Brian Shin (BS)
Nick Hom (NH)

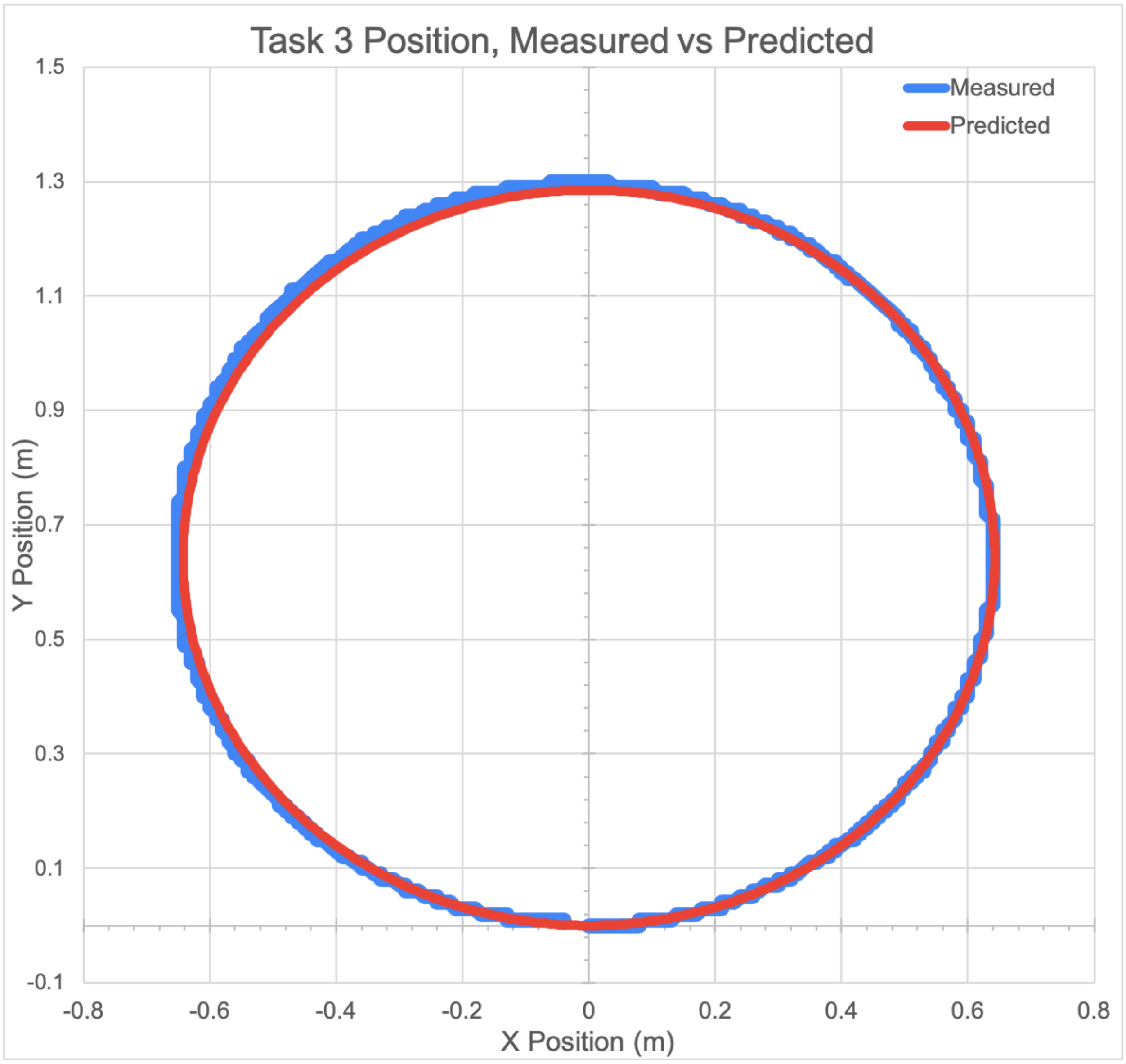

Figure 3: x and y position, both measured and predicted of the Romi driving a curved path for 30 seconds, with the intention of driving a full circle

c. R Value Estimate
   i. The R value, the distance from the center of the robot to the Instantaneous Center of Curvature or in this case the radius of the circle, of this curved motion was about 0.65 m.
   ii. This matches very well with both the predicted value of 0.6425 m and the measured value of 0.645 m (estimated based on max y value).

4. **Final Pose Error:**

a. Task 1
    i. In this task, the final pose of the robot was (0.01m, -0.00m, -3.14 rad). The final pose that the robot was supposed to achieve was (0m, 0m, -3.14 rad) since the task was to drive straight, turn 180 degrees (or π radians), then drive back. The error in this experiment is 0.01m in the x direction, which is pretty accurate considering the inherent inaccuracies of odometry.

b. Task 2
    i. In this task, the final pose of the robot was (0.46m, 0.29m, 3.41 rad). The final pose that the robot was supposed to achieve after driving straight and moving in a curved trajectory for each 10 seconds was (0.45m, 0.28m, 3.48 rad). The error here was 0.01m in both x and y directions and 0.07 radians in the orientation. This is not as accurate Task 1, but pretty close taking into consideration the complex curve the robot followed.

c. Task 3
    i. In this task, the final pose of the robot was (-0.04m, 0.01m, 6.25 rad). The final pose that the robot was supposed to achieve was (0m, 0m, 6.28 rad) after driving a full circle - doing so should result back to the starting position but with a full 2π radian heading. This experiment did not yield as accurate results as Task 1 but still was reasonably close especially considering the total distance traveled, speed traveled at, and total travelling time.

d. Across all tasks, the main cause for error in the experimental paths was likely the discrepancy between the given/measured L value - the wheel track - and the actual wheel track that the robot operates with. The given value was 0.142875 m, however measured on Matt's robot it was between 5-10 mm less, and on Brian's robot, about 2mm less. This is due to the design and random variances between every manufactured Romi. The axles by design are also cantilevered and the wheels are not well axially constrained to such axles. With this, the wheels can angle upward or even wobble during driving, causing the real contact patches on the wheels to not be accurately reflected in the measured or given wheel track values. Thus, the calculations involved with predicting the position of the robot and other similar operations will have errors compared to the experimental paths. The wheel track also would change based on the motion the robot was performing, it seemed to get narrower as the difference in the wheel speeds increased making the measurement error change as the path changed.

5. **Task 3 but with a Different Value for L**
    a. The speeds were the same as for Task 3 (VL = 120 mm/s and VR = 150 mm/s)
    b. Plot

Figure 4:

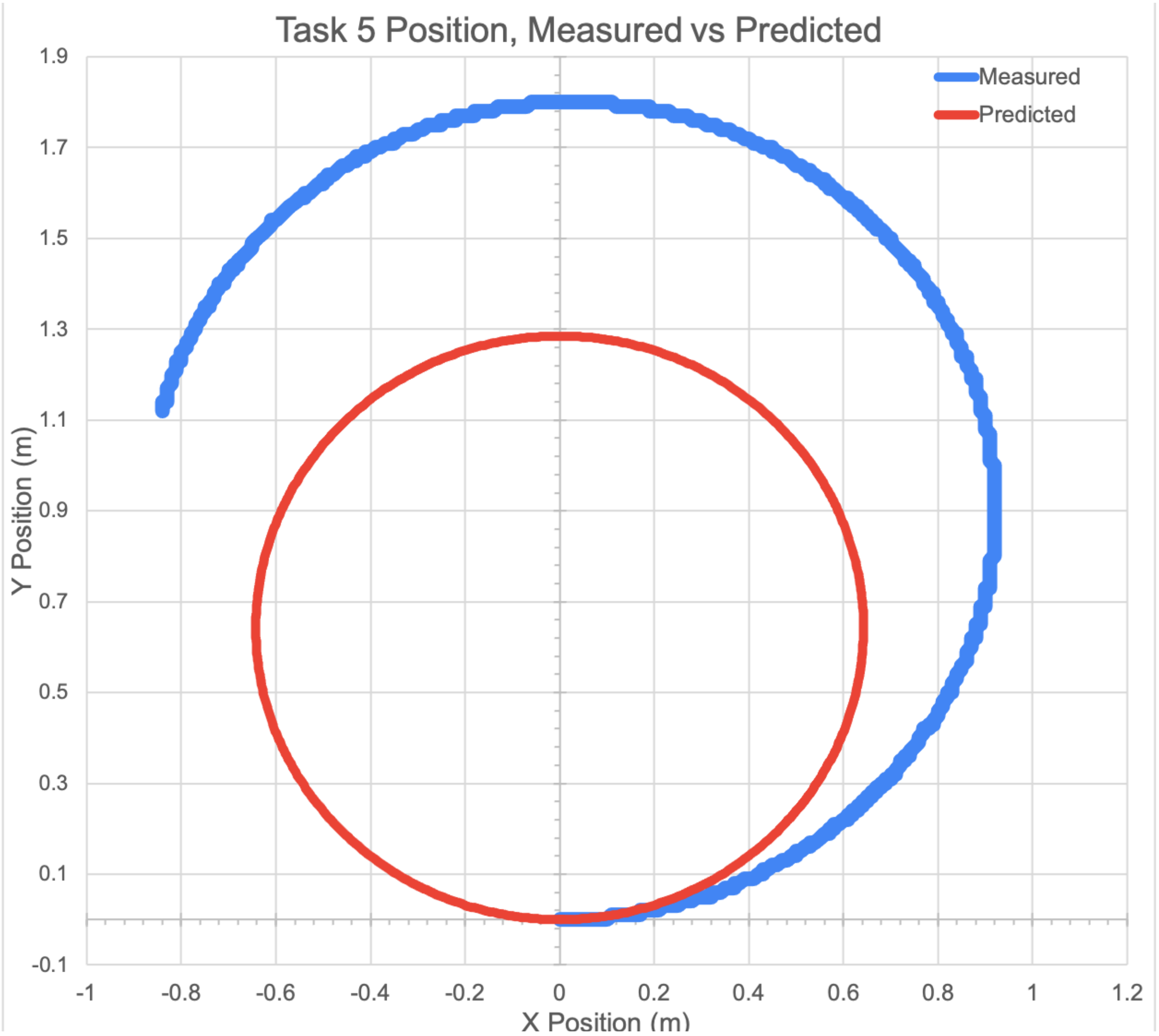

Figure 4: x and y position, both measured and predicted of the Romi driving a curved path for 30 seconds, with the intention of driving a full circle. Same path as Figure 3 with a different value of L inside Position_estimation.h

c. Differences
   i. By making the value of L used in the position estimation code greater than the value to actually described the robot makes the measured quantity of ω, the angular velocity, or rate of change of θ, the heading, smaller which combined with the similar increase in R, or the distance to the ICC, makes the path travelled no longer be a complete circle, but an incomplete one with a larger radius, even if that no longer accurately describes the motion of the robot. As shown in Figure 4, the two curves are not remotely the same, even though they are supposed to describe

the same path. This shows one of the issues with odometry in that it requires accurate measurements of the robot and when trying to apply those details to a fleet of robots, manufacturing differences can cause inconsistencies between the behaviors and measured data from different robots.

6. **Smallest Possible Measureable θ**
    a. The resolution on the Romi wheel encoders is 1440 counts (or ticks) per revolution of the wheel. This number is inclusive of the 120:1 reduction gearbox. As for the smallest possible change in heading (θ) that can be measured by the encoders, this relies on the radius of the wheel and the wheel track of the robot. In this case, we will assume a wheel radius of 35mm and a wheel track of 140mm. These measurements are very close to the actual values from the Romi and make the math much more simple. The circumference of the wheel is responsible for moving the robot, and thus each wheel revolution can move the robot $2\pi * 35mm$, or 219.9mm. One count of the encoder equates to a distance of $\frac{2\pi * 35mm}{1440}$ mm or 0.1527 mm. In a point turn (center of the wheel track is the point of rotation), this distance represents an arc length of a circle that the robot is tracing, with a radius of half the wheel track. Using the arc length equation $s = r\theta$, we can find that the angle in radians from this movement is $\frac{0.1527\ mm}{70mm}$ or 0.002182 radians. A quick conversion brings us to 0.125 degrees - the smallest measurable heading of the Romi with the encoders.

7. **Sensor Fusion**
    a. Sensor fusion is defined as processing input across several sensors and using special algorithms to filter the sensor data to create an output that is significantly more comprehensive than any individual sensor. If done well, the inaccuracies of the individual sensors will be eliminated. As for position estimation on the Romi robots, this concept certainly improves the accuracy and reliability of position estimation for kinematics. If one were to simply use the wheel encoders alone to track position, there are a number of errors that can occur, such as wheel slippage or environmental factors that would lead to inaccuracy in the position estimate. By combining another sensor, the IMU, with the use of the wheel encoders, a more accurate position estimate can be achieved. The accelerations and angular velocities read from the IMU can be derived into velocity and position measurements that can supplement the wheel encoder data. Beyond these two sensors, several other sensors can be used in conjunction to create a much more precise measurement of position, such as distance sensors (when appropriate), IR beacons, or camera vision (with physical mapping like AprilTags).

Team 10:
Matthew Haahr (MH)
Brian Shin (BS)
Nick Hom (NH)

8. **Determining the position of a robot using wheel encoders and an IMU is prone to error. What are commonly used methods to compensate for those errors that accumulate over time?**
    a. External position detection using beacons like with triangulation or trilateration. By using these sensors and external beacons, they eliminate any accumulated error that exists with on-board sensors as these external sensors are not moving and have absolute positions that can be relied on.
        i. Triangulation uses the measured from the origin to known positions and derives the position of the robot by using those measurements and trigonometry.
        ii. Trilateration uses measured distance to three or more known positions to calculate a distance. GPS uses trilateration by calculating the time to the three nearest GPS satellites and then solving for the position
    b. Landmark and mapping: similar to beacons, using landmarks and other forms of mapping eliminate the error that comes with using on-board sensors. Landmarks and mapping again have absolute positions and do not usually move so these can be relied on to interpolate the position of a robot. When doing landmark style navigation, there are two common forms: Natural and Artificial. Natural Landmarks utilize already existing objects like statues, trees, and buildings to navigate a space. Artificial Landmarks consist of adding objects to the environment to allow the robot to navigate. An example of this is placing AprilTags on walls to have the robot follow a set path.
    c. AprilTags and overhead position mapping: For overhead position mapping, the experimental or observed path of the robot is often accomplished with a combination of easy visual identifiers (like AprilTags) or reflective tape and a camera system above. By doing so, the robot is tracked on a coordinate system through the identifier and the absolute position can be obtained relatively easily. This is external to the robot and eliminates any internal error accumulation that is inherent to on-board sensors. Additionally, the overhead position mapping can be used in conjunction with collision detection on the robot to further any data on physical boundaries.