Please Edit this page with your details

Introduction

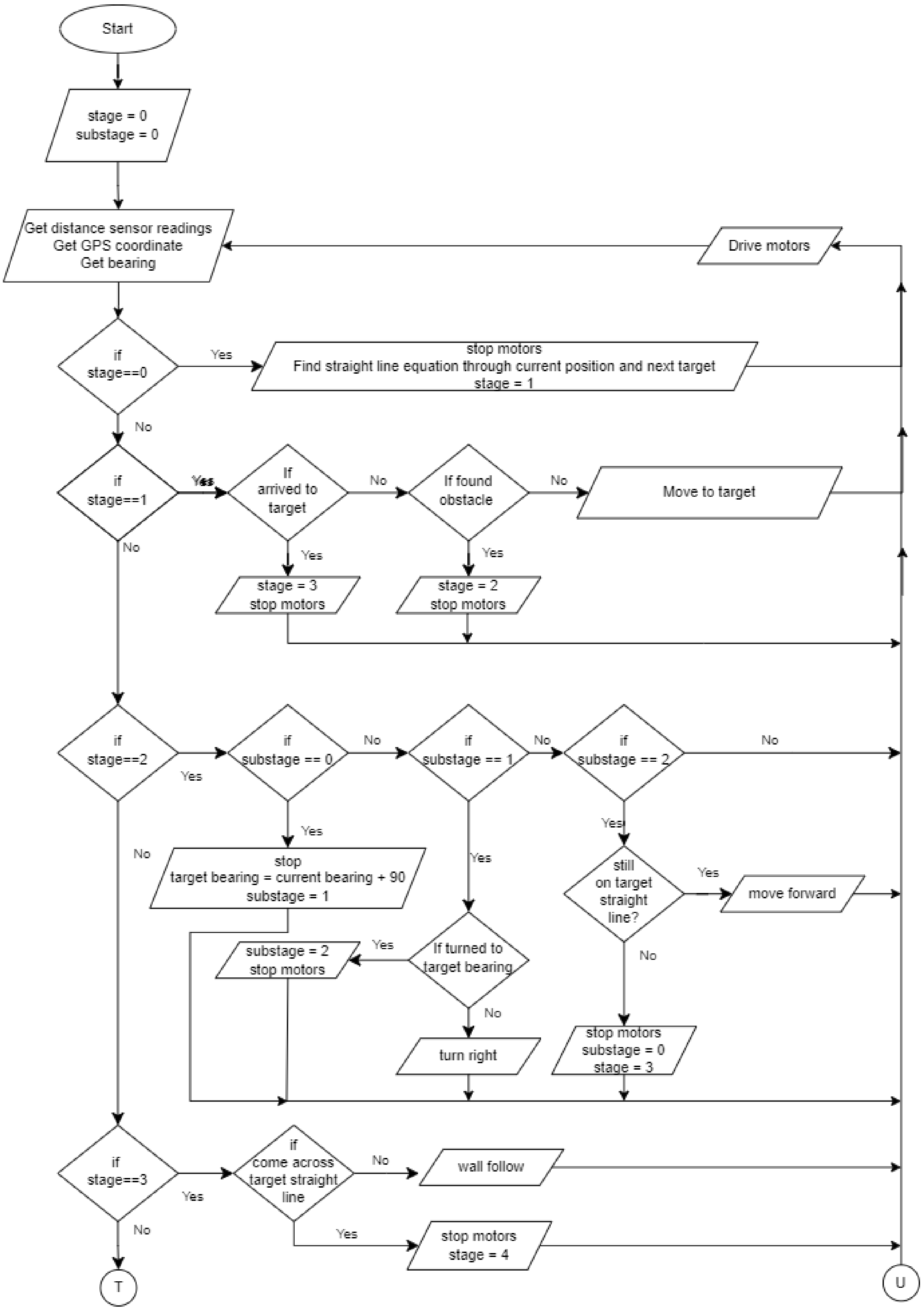
In this simulation, two autonomous robots operate independently: a stationary manipulator with 3 degrees of freedom (RRR) and a mobile robot. The mobile robot navigates to the stationary robot while avoiding obstacles, waits for the manipulator to load a package onto it upon detection, and then proceeds to the delivery point, again navigating around obstacles. This process entails the mobile robot autonomously traveling to the stationary robot, receiving the package, and navigating to the delivery point, all while ensuring obstacle avoidance throughout.

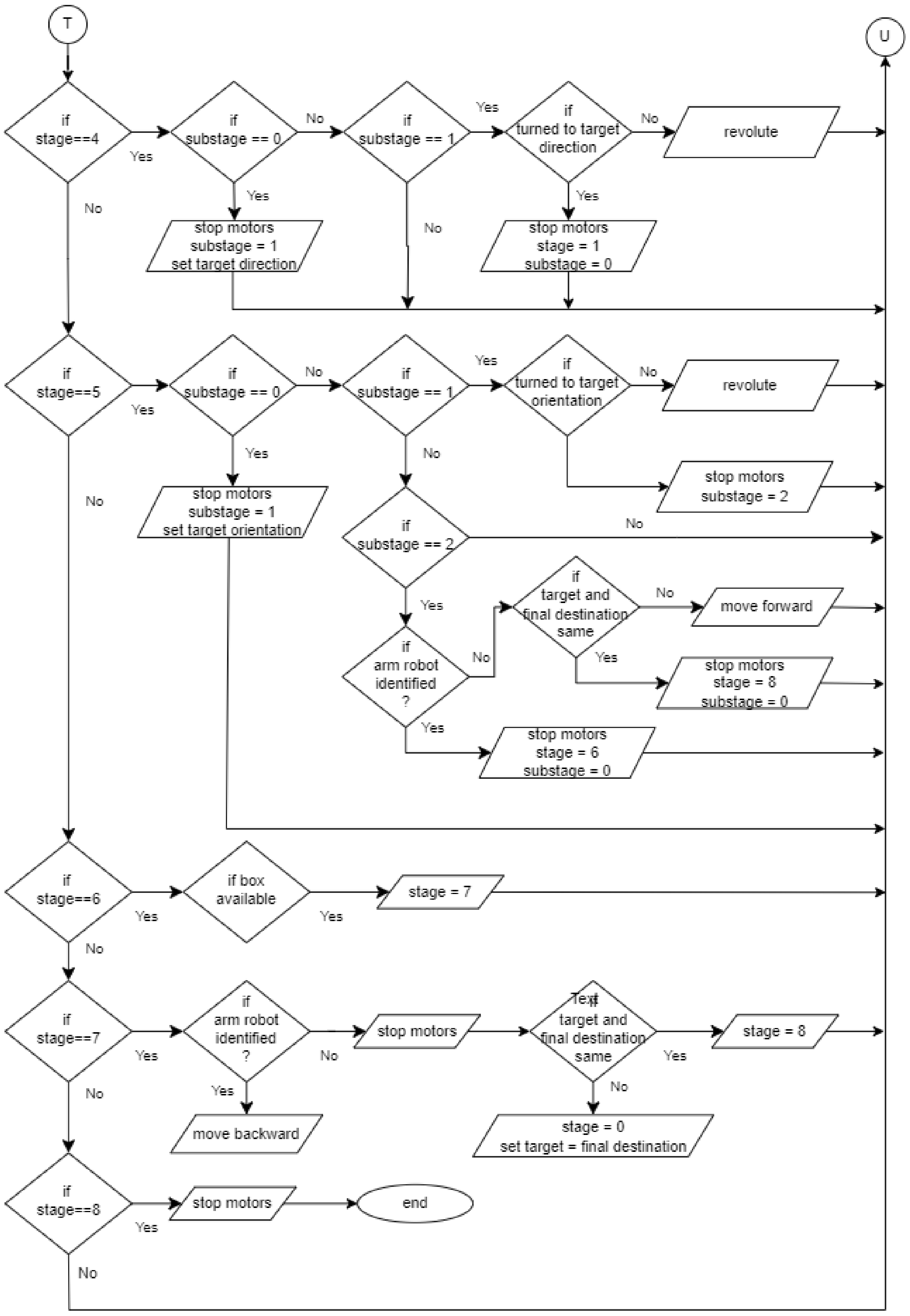
Mobile Robot

The mobile robot has been engineered to fulfill specified criteria, boasting capabilities such as autonomous navigation to predetermined destinations while adeptly avoiding obstacles. Its design includes a 4-wheel-drive (4WD) system for enhanced maneuverability. Obstacle avoidance is accomplished by integrating bug algorithm techniques with wall following algorithms, ensuring efficient traversal even in cluttered environments. Additionally, the robot achieves localization through a combination of GPS readings and bearing calculations via a compass, allowing for precise orientation adjustments as needed.

- Three distance sensors are utilized for obstacle detection, meeting the specified criteria
- On the top bed of the robot, two distance sensors are employed for load identification. Originally intended to utilize touch sensors, a flaw in the WEBOTS environment led to their failure. Instead, two distance sensors are configured in a two-ray setup, strategically positioned to ensure at least one ray intersects with the load
- To facilitate wall following, two distance sensors are deployed. Adhering to the requirement of always turning right upon obstacle detection, left wall following is implemented. Hence, the sensors are positioned on the left side of the robot
- Localization is achieved through the use of GPS sensors and a Compass

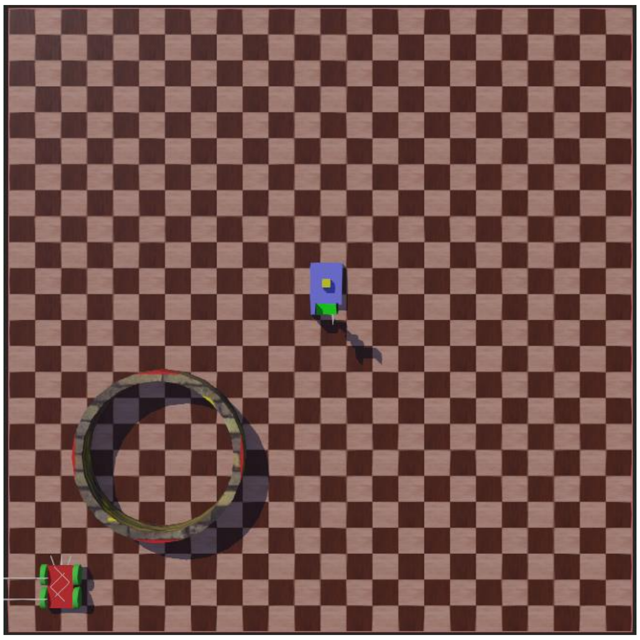
Controller algorithm flow chart

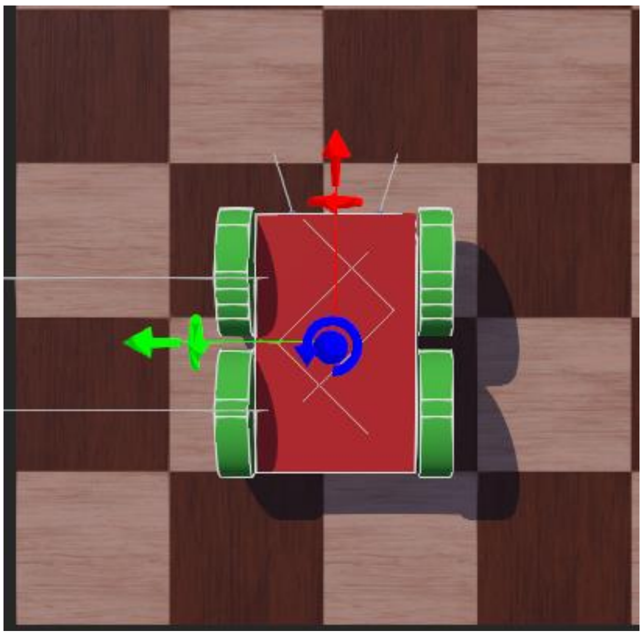



T
U
if stage==4
Yes
No
if substage == 0
No
Yes
stop motors substage = 1 set target direction
if substage == 1
Yes
No
if turned to target direction
No
Yes
revolute
stop motors stage = 1 substage = 0
if stage==5
Yes
No
if substage == 0
No
Yes
stop motors substage = 1 set target orientation
if substage == 1
Yes
No
if turned to target orientation
No
revolute
stop motors substage = 2
if substage == 2
No
Yes
if arm robot identified ?
No
Yes
if target and final destination same
No
Yes
move forward
stop motors stage = 8 substage = 0
stop motors stage = 6 substage = 0
if stage==6
Yes
No
if box available
Yes
stage = 7
if stage==7
Yes
No
if arm robot identified ?
No
Yes
move backward
stop motors
Text
if target and final destination same
Yes
No
stage = 8
stage = 0 set target = final destination
if stage==8
Yes
No
stop motors
end

Navigation algorithm

The robot's localization is accomplished by utilizing GPS coordinates provided by the GPS sensor, while its heading is determined through the compass sensor, with the resulting value converted into a bearing relative to the North direction indicated by the green arrow in the top view of the arena. This integration of GPS and compass data enables precise localization and orientation of the robot within the arena environment. Before starting on its journey, the controller computes the equation of the straight line connecting the starting point and the destination point, while also determining the bearing of the destination relative to the starting point. Consequently, the robot consistently endeavors to follow this straight path. Upon encountering an obstacle, the robot executes a right turn and engages in wall following around the obstacle until it intersects the straight line once again. Subsequently, the robot resumes its course along the straight line towards the destination.


Top view of arena


Top view of mobile robot

Obstacle avoiding algorithm

Upon encountering an obstacle along its path, the robot executes a 90-degree right turn from its current bearing. Subsequently, it initiates wall following along the obstacle's perimeter until it intersects its original path. Once the original path is rediscovered, the robot resumes its trajectory along this predefined route towards its destination. Following equations show, how the speed values for wall following are calculated based in left side distance sensors.

- a is the distance between two distance sensors
- dRL and dFL are rear left distance sensor reading and front left distance sensor reading respectively.
- dWallSide is the expected distance to be maintained with robot and wall
- kWall is gain which is tuned by trial and error method.
- v maximum allowed linear velocity of the robot
- r is the wheel radius
- b is the half the width of the robot

- e distance from rear to axle.

$$\emptyset = \tan^{-1}\left(\frac{dRL - dFL}{a}\right)$$

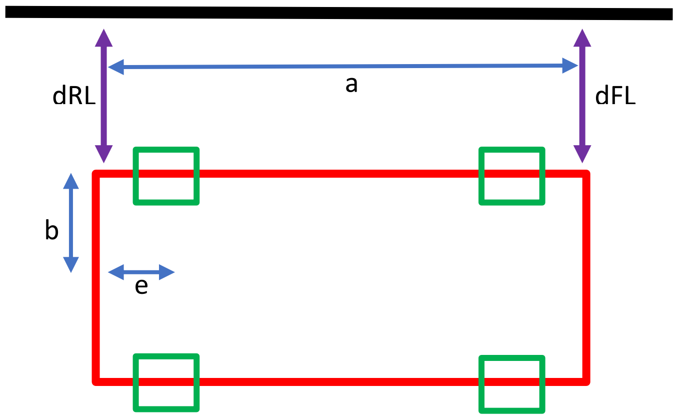
$$d = \frac{dRL + dFL}{2} - dWallSide$$

$$\gamma = kWall * d$$

$$\alpha = \emptyset + \gamma$$

$$wL = \frac{v}{r}\left(\cos(\alpha) + \left(\frac{b}{e}\right) * \sin(\alpha)\right)$$

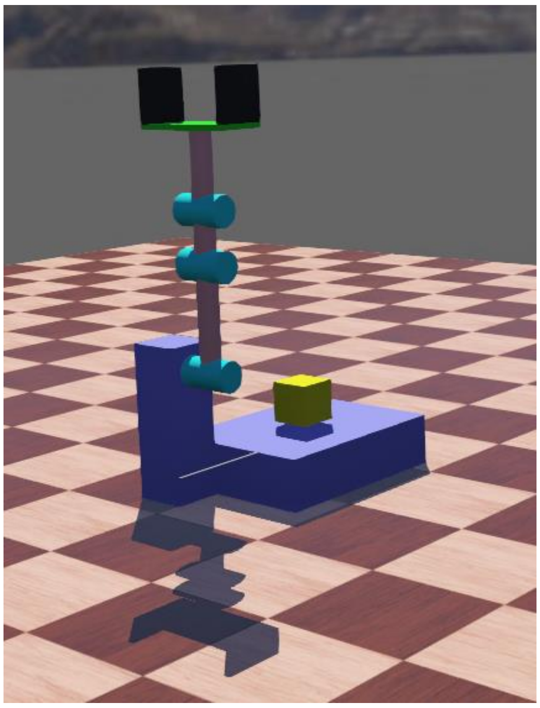
$$wR = \frac{v}{r}\left(\cos(\alpha) - \left(\frac{b}{e}\right) * \sin(\alpha)\right)$$



Mobile robot wall follow diagram

Manipulator Robot

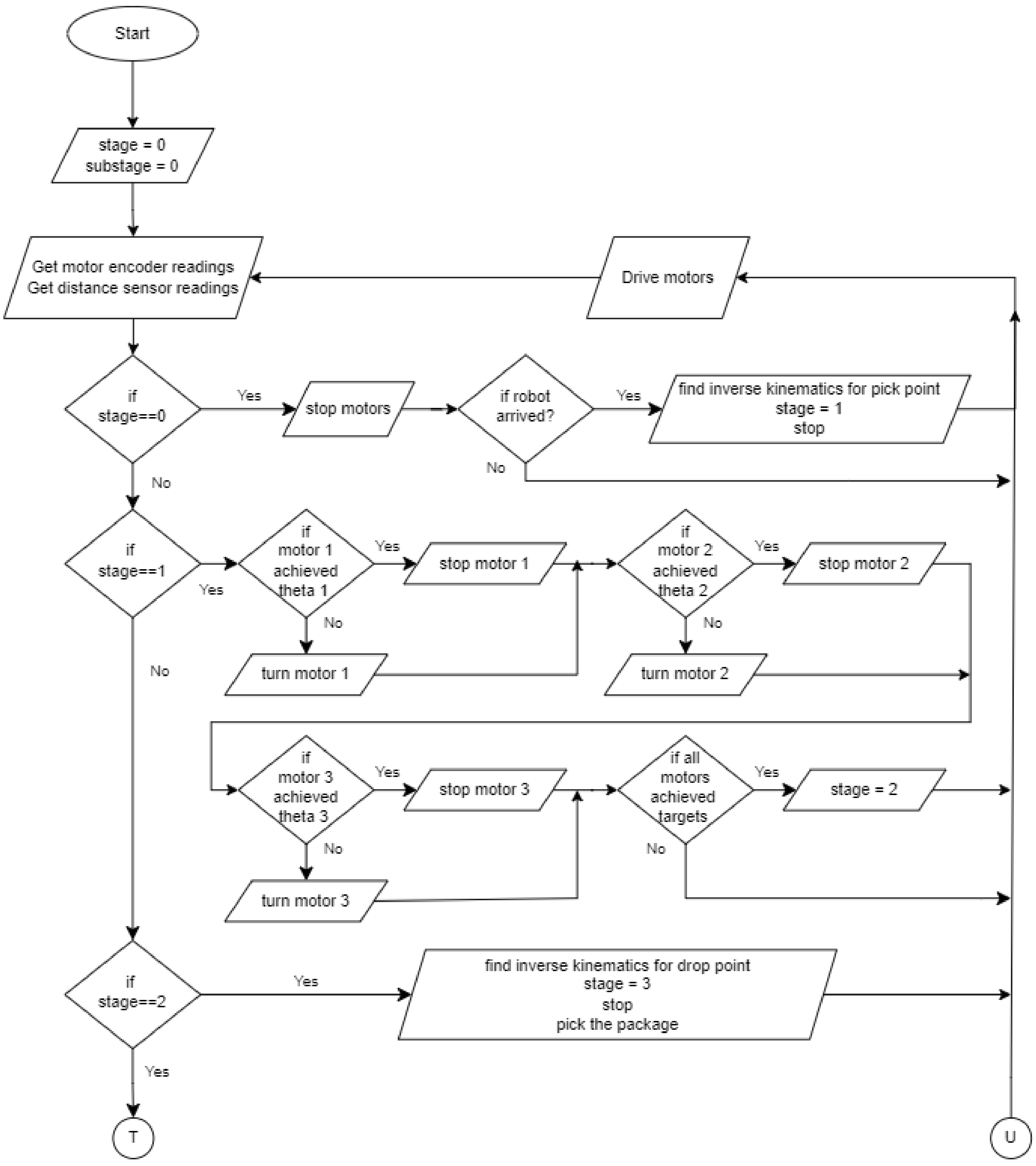
The manipulator robot is designed to meet specified requirements, including precise link lengths. It possesses the capability to compute inverse kinematics for points located within its workspace. This ensures the robot's adeptness in accurately determining joint configurations necessary to reach designated points, facilitating efficient manipulation tasks as per the provided specifications.

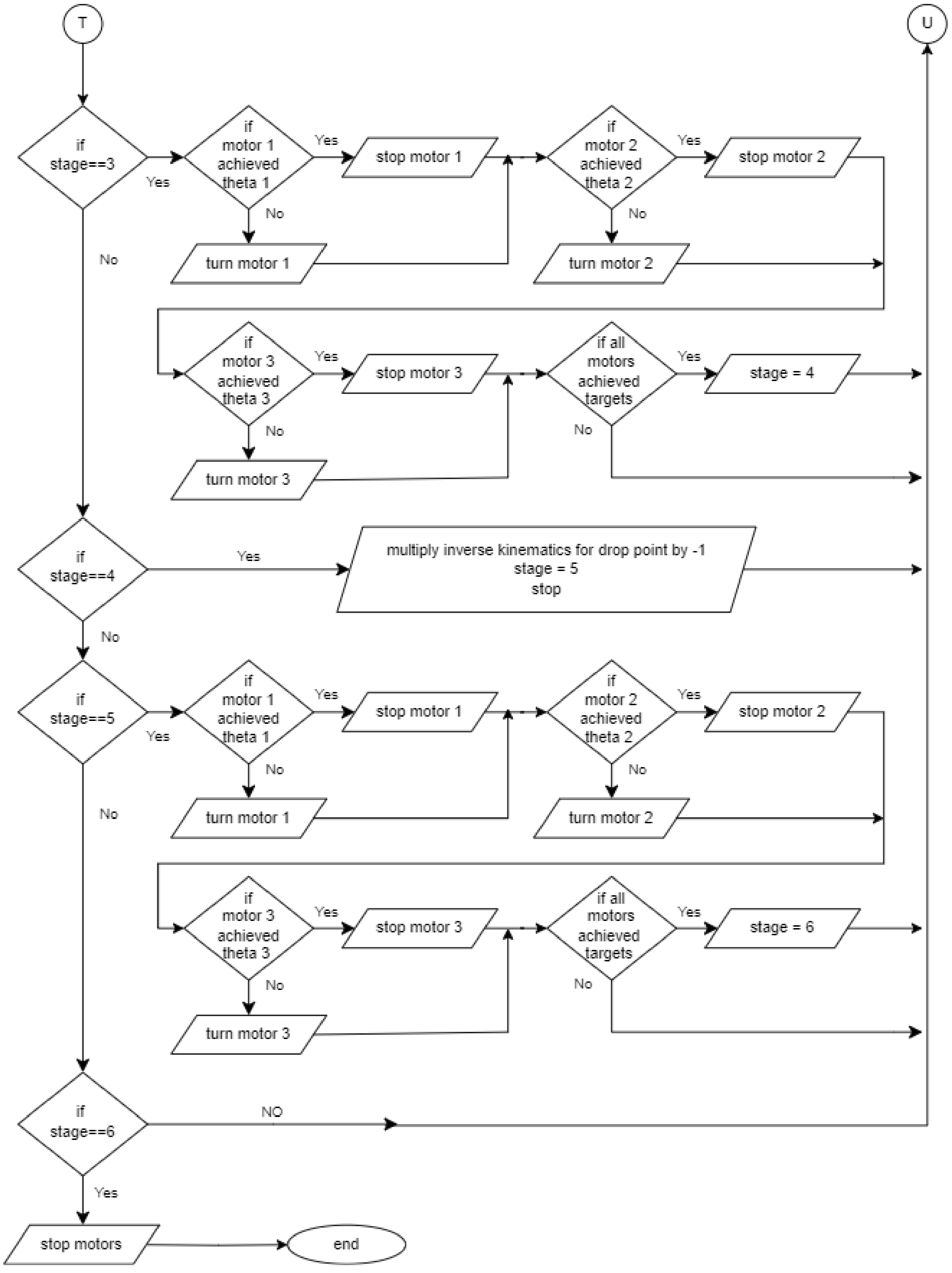

Manipulator robot initial position

The manipulator remains stationary until the mobile robot arrives at its designated location. Once the mobile robot is in position, the end effector of the manipulator follows a pre-planned trajectory, encompassing pick point, drop point, and negative of pick point. To optimize efficiency and ensure obstacle-free paths, the trajectory of the end effector is planned in joint space. The gripper is oriented to seamlessly slide into the package without collision, facilitated by symmetric type connector nodes embedded in both the end effector and the package. The manipulator executes a sequence of movements: approaching the pick point, gripping the package, moving to the drop point to load the package onto the mobile robot, and finally returning to the negative of the drop point.

- A distance sensor is employed to detect the arrival of the mobile robot
- Encoders are utilized to accurately read the position of the motor shaft

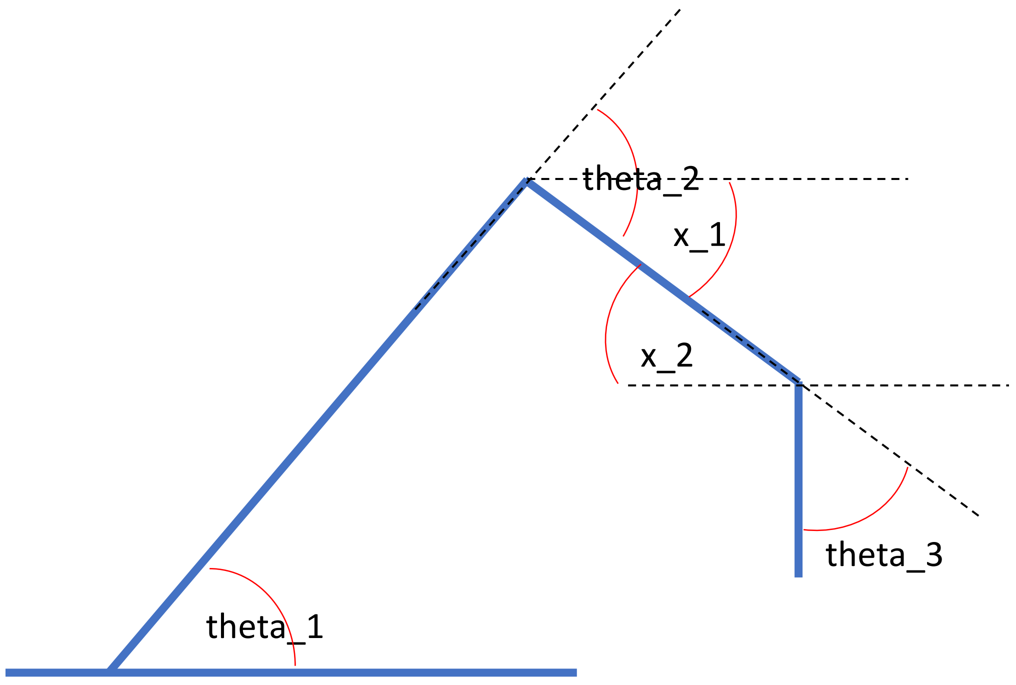
Controller algorithm flow chart




T
U
if stage==3
Yes
No
if motor 1 achieved theta 1
Yes
No
stop motor 1
turn motor 1
if motor 2 achieved theta 2
Yes
No
stop motor 2
turn motor 2
if motor 3 achieved theta 3
Yes
No
stop motor 3
turn motor 3
if all motors achieved targets
Yes
No
stage = 4
if stage==4
Yes
No
multiply inverse kinematics for drop point by -1
stage = 5
stop
if stage==5
Yes
No
if motor 1 achieved theta 1
stop motor 1
turn motor 1
if motor 2 achieved theta 2
stop motor 2
turn motor 2
if motor 3 achieved theta 3
stop motor 3
turn motor 3
if all motors achieved targets
stage = 6
if stage==6
NO
Yes
stop motors
end

Inverse kinematics

Both the pick point and drop point require the final link of the manipulator to be vertical, effectively reducing the system to a 2-degree-of-freedom (2-DOF) configuration. In this setup, the y-coordinate is determined by subtracting the length of the final link, while the x-coordinate remains constant. Using trigonometry, two simultaneous equations are derived in terms of theta_1 and theta_2. These equations enable the calculation of theta_1 and theta_2, thereby determining the manipulator's positioning. Additionally, theta_3 is obtained through basic triangular geometry principles. Further details on this methodology are elaborated in a MATLAB article referenced in the supplementary materials.



When above diagram is considered,

- x_1 is (theta_2-theta_1)
- x_2 become equal to x_1 as both the angles are alternate angles between two parallel lines.
- Finally the sum of angles x_2 right angle and theta_3 forms a straight line.
- Therefore theta_3 is (π/2)-x_2

Problems faced

- Compared to the dimensions of mobile robot, dimensions of manipulator are relatively small, which makes difficult to insert the drop point in the workspace of the manipulator.

- Touch sensors were to be employed in package detection in mobile robot and detecting arrival of mobile robot in manipulator robot. But due to flaw in WEBOTS environment, touch sensors were not accurate.

Referrences

Matlab article of 2 dof manipulator inverse kinematics

https://www.mathworks.com/help/symbolic/derive-and-apply-inverse-kinematics-to-robot-arm.html

Webots documentation

https://cyberbotics.com/doc/reference/nodes-and-api-functions

Webots robot arm tutorial and encoder tutorial

https://youtu.be/Tm4AiS0AY3s?si=aJXEFoWP7440-CpJ

https://youtu.be/BifEa-R_rEI?si=LUmrFPllf0Wp4UIc

Wall follow algorithm and BUG algorithm by Senior lecturer Leopoldo Armesto

https://youtu.be/DkOXpZtEZwg

https://youtu.be/HmkX9aiWISU