Asstive Robotics: Alleye

Josh Field, Blake McHale, James Tyler & Theodore Lutkus
EECE 5552: Professor Ramezani

1. Introduction

Alleye is an all purpose guidance system meant to aid blind and visually impaired individuals. Visual impairment is a significant problem around the world with around 36 million people being affected. Alleye provides an affordable alternative by replacing guide dogs with an overhead unmanned aerial vehicle (UAV). This vehicle recognizes obstacles in the way of the user and generates a path that avoids these hazards. The user then follows the path via tactile feedback from a guidance device. This path is also updated dynamically as obstacles move in the scene, so the user is never put in harms way. These steps are visualized below in Figure 1.

Figure 1. Operational View Diagram of Alleye.

2. Approach

In order to address this problem we chose to limit the scope for different aspects of the project. The UAV is only implemented in software, while a prototype for the tactile guidance system was developed as a proof of concept.

2.1. Simulation

We chose to develop the project with ROS Noetic in Ubuntu 20.04 so that we could build an environment that contained obstacles and leverage open source ROS packages for each aspect. In Gazebo, the simulation environment, we created a simplified crosswalk world (Figure 2). Default gazebo models were used for all of the obstacles, and were placed to present a challenging path planning problem.

To control the UAV in the simulation we extended the interface from the SJTU drone package [2]. Specifically, the UAV was simply able to take off and hover above the crosswalk.

Figure 2. Gazebo crosswalk environment.

2.2. April Tags

The use of April Tags is another result of our goal to simplify the scope of the project. An additional simplification was the assumption that all relevant tags would be visible in the frame of the stationary camera. April Tags are fiducial markers which we used to gain simple and accurate poses of the client, goal, and obstacles in the simulation environment. The apriltag_ros package was used by simply adding it to our launch file and it published all detections to the "/tag_detections" topic. In order to differentiate between the client, goal, and obstacles we simply created a config yaml file that allowed us to assign specific tag IDs respectively. Since the April Tags only represented a single point, we also added bounding box parameters to the config yaml file that captured the relative size of the obstacle.

2.3. A Star Path Planning

The path planning algorithm implemented with ROS and Gazebo was A Star. This algorithm was chosen since it is popular and efficient for 2D path planning. The implementation used was from this Github [1]. The path planning algorithm was integrated with ROS through Python.

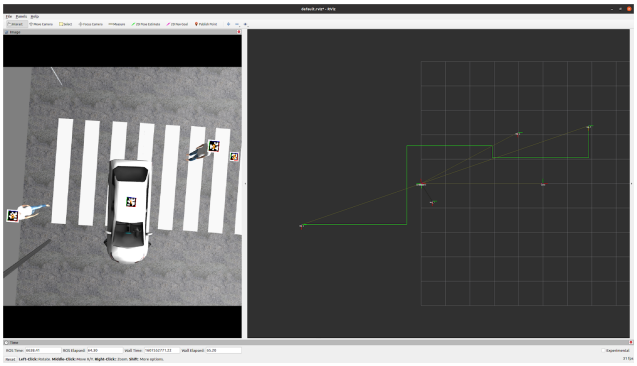

Figure 3. AprilTags and A Star path in RVIZ.

ROS was used to grab information about the apriltag positions relative to the drone and the world through the tf library. These positions were then remapped, so they matched the positions of the discretized grid that A Star was being performed on. This solution is fully configurable through a yaml file that specified the boundaries of the world and the scale of discretization. After converting the positions to the grid world representation, an occupancy array of 1s for objects and 0s for free space was created. This array representation was used with A Star to generate a path from a starting location to goal location. Once this path was generated it was transformed back to the world frame and published as a ROS Path message. Figure 3 shows the generated path in green. It can clearly be seen that the path moves around the car and person as it reaches the goal apriltag on the far right.

Additionally, the code that was implemented creates a ROS node that constantly runs the A Star algorithm such that the resulting path can be updated dynamically as apriltags move in the environment.

2.4. Probabilistic Foam Method

A Star is a simple, computationally cheap path planning algorithm, but it can create paths that are difficult to follow or potentially dangerous. When used for assistive applications, path planning needs to be smooth, and it needs to provide a safe buffer for the user.

The Probabilistic Foam Method, presented by Nascimento, et al. attempts to fulfill both of these goals [3].

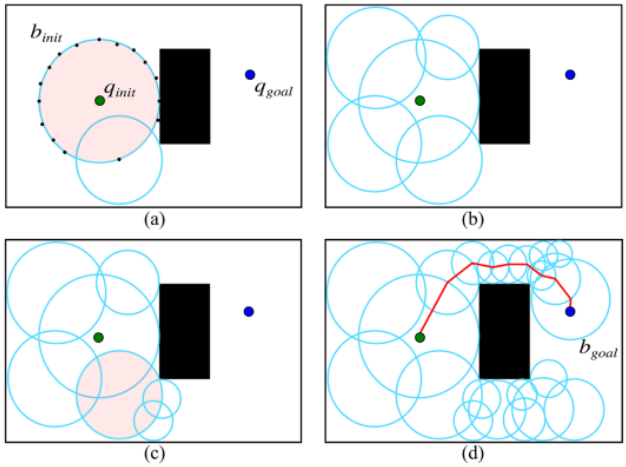

Figure 4. PFM Path Propagation

2.4.1 Foam Propagation

The Probabilistic Foam Method uses a series of bubbles to create a safe path along which the user can travel. PFM begins by expanding a circle around the initial point until the circle contacts an obstacle. Then, uniformly spaced, random points, are selected along the radius of the circle.

These points acts as the centers of ”child” circles. Additionally, every point that is encircled by a child circle is removed. Figure 4.a shows the creation of the initial bubble and the first ”child” bubble. In Figure 4.b, the same method is continued for all remaining points on the circumference of the initial bubble. Figure 4.c shows the propagation of the foam on each of the child bubbles. This propagation continues until a child circle encompasses the goal point, as shown in Figure 4.d.

2.4.2 Alleye Application

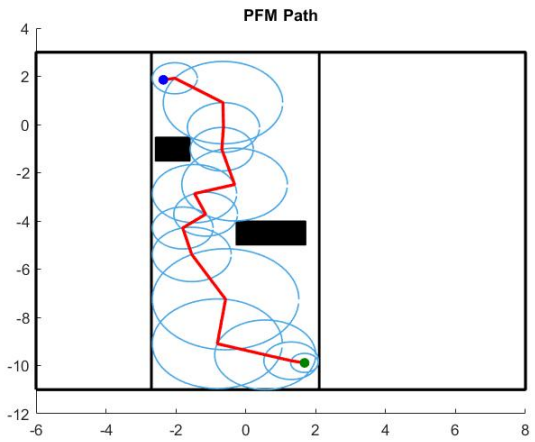

Figure 5. PFM Applied to the Crosswalk Problem

In the scope of this project, PFM was implemented as a proof of concept, not in real-time. Figure 5 shows the PFM planned path for the Alleye environment.

The Probabilistic Foam Method demonstrates several limitations for our application. First, it is significantly more computationally intense. It would be difficult to perform real time path planning at a high frequency with PFM, certainly in a power constrained environment. Additionally, in a space with many obstacles, the path created by PFM is prone to many rapid, sharp turns, which are not ideal for an assistive technology.

2.5. Tactile Guidance

The previous discussion on simulation and algorithmic path planning demonstrates that our work can observe the world and make decisions for our end user to navigate crosswalks. To provide the end user with input to guide them along an optimal path, we developed a mechanical device that translates navigation commands into tactile guidance for the user.

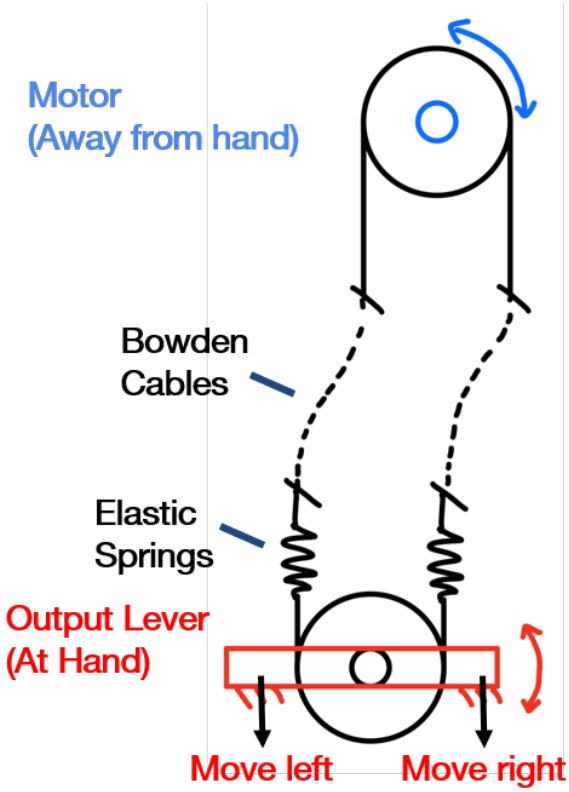

Figure 6. Series Elastic Actuator Concept

The guidance device consists of a servo motor for actuation, bowden cables for flexible power transmission, series elastic springs for natural compliance, and an output lever attached to the end user, as illustrated in Figure 6. The bowden cables in this design allow for the servo motor to be located away from the user's hand, which assists with weight and will keep the actuator's electronics dry when raining. The series elastic springs provide compliance to the system such that the user is still able to freely move their hand without risk of injury from the servo's motion.

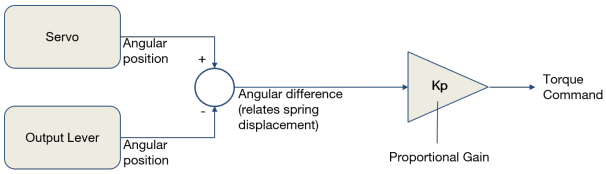

Figure 7. Tactical Feedback Device Controller

To apply controlled torque signals to the user's hand, our team leveraged the physics of our series elastic springs to avoid the need for expensive torque sensing. We first approximate the servo's angular position to be equal to its PWM commanded angular position for slow movements. With an angle sensor at the output lever, we can compare the output's rotation to the input to calculate the displacement experienced by the elastic springs. Using Hooke's law, the force exerted by the springs can be estimated with known spring constants and the spring displacement. This force is then converted to torque with the known radius of the output pulley. The resulting system relating input angular position to output torque is first order and therefore a simple proportional controller was used to apply desired output torque.

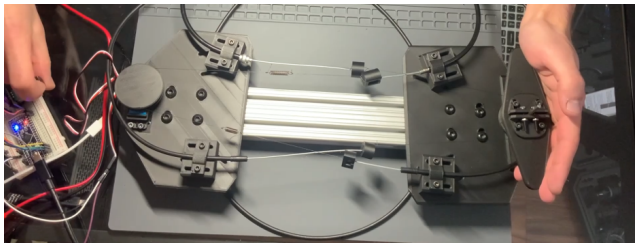

Figure 8. Tactical Feedback Device

To test the tactile guidance device, a mechanical system was constructed and the torque controller was embedded in firmware on a microcontroller. Although the input and output of this device are shown attached to a single metal link for testing, the bowden cable design would allows for the input and output to be located apart from each other. In testing, the system was successfully able to prescribe torque commands at the output. The produced mechanical device demonstrates that this low cost, compact tactile guidance system could be utilized in a real-world application.

3. Conclusion & Future Work

Through this project we successfully developed a proof of concept for the Alleye system through the development of a UAV simulation, ROS software control for dynamic

path planning, and a physical prototype for the tactile guidance of the visually impaired. The software developed is available at our Github. Future work to make this project a reality would need to include expanding the scope to build a more robust system. This would include a more complex simulation environment with dynamic obstacles and a moving UAV. Improved path planning could create a safer, smoother route. The application of modern Computer Vision techniques such as object recognition could be utilized to detect the positions of obstacles and the client in the world. The final improvement would be to build a physical UAV platform and to test the entire system in the real world.

References

[1] A. Dahdouh, https://github.com/atomoclast/realitybytes_blogposts. Publsihed 2018. Accessed December 18, 2020.

[2] T. Köse, https://github.com/tahsinkose/sjtu-drone

[3] Nascimento, L.B.P., Barrios-Aranibar, D., Alsina, P.J. et al. A Smooth and Safe Path Planning for an Active Lower Limb Exoskeleton. J Intell Robot Syst 99, 535–553 (2020).