🔥 RescueBot 🔥

Eli Gooding[#1], Isaac Tong[#2], Justin Ng[#3], Jubilee Wang[#4]

[1]eliwg@umich.edu [2]isaacto@umich.edu [3]justinjn@umich.edu [4]jubileew@umich.edu

***Abstract*— Our team created a fire detection robot with search and rescue capabilities. The robot is able to create a heat mapping of a burning building, identify a person, and lead that person to safety.**

***Keywords*— search and rescue, IR sensor, thermal imagery, path finding, fire detection**

Introduction

Emergency response environments are uncertain, challenging and dangerous for rescue teams. For instance, firefighters do not know which rooms and hallways have active fires or are safe to enter. To assist first responders, we have designed an autonomous robot to map out fires in a burning building to give rescue teams better insight into the risks and hazards of their operational environments. In addition, our robot can be used in search and rescue missions to augment the efforts of human rescuers..

Related Work

Similar products that conduct search and rescue operations autonomously include the Boston Dynamics robot dog–the Spot dog–that emulates a real dog to assist firefighters. The Spot dog detects and navigates obstacles and collects data on structural integrity of a collapsed building, as well as measures toxic gasses. [1]

In addition to more traditional forms, robots emulating snakes and other animals are in the process of being created to travel through smaller spaces to perform search-and-rescue operations. [2]

A team from UCSD developed a “firefighting robot” that generates a 3D point cloud with thermal overlays using a stereo and depth camera. [3]

Similarly to these products, RescueBot will be able to enter dangerous areas and assist first responders. RescueBot is differentiated in its ability to lead survivors to safety following identification. In addition, it uses LiDAR for mapping and localization.

Methodology

RescueBot, shown in Figure 1, utilizes a combination of sensor readings from its LiDAR, IR Thermal Camera, and RealSense depth camera to create a thermal heat map of its surroundings. SLAM is implemented through using the LiDAR and heat mapping is implemented through fusion of sensor data from both the depth camera and IR Thermal camera. This methodology section will explain in detail how we were able to implement the two mapping and search-and-rescue modes through using various camera calibration techniques, pose estimation, modifications to the motion controller and BotGui. Our approach required the use of the following materials:

Bill of Materials

- 2 Mbot-Mini robots
- MLX90640 55° IR Thermal Imaging Camera
- Intel RealSense Stereo Depth Camera D457
- Tea light candles
- Aluminum Soda Cans

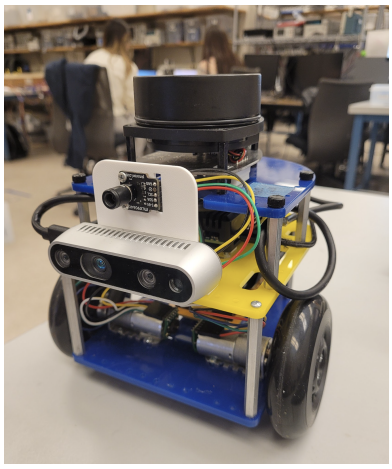

Figure 1: RescueBot with LIDAR sensor, thermal camera, and RealSense depth camera mounted

Multi-camera calibration requires both cameras to always be at the same relative position, therefore we found it necessary to 3D-print a mount so that both cameras are fixed securely. This would prevent error

due to shifting in the fields of view of the cameras as well as remove the need to constantly recalibrate the cameras. Using SolidWorks, we designed a mount that would allow us to securely mount the depth and thermal cameras on the RescueBot similar to the mount for the pre-installed camera on the MBot. Following the dimensions of each camera, we created cutouts for where each camera would be screwed on and for threading soldered wires through. The base has two holes where it can be attached to the robot. The attached mount can be seen in Figure 2 below.

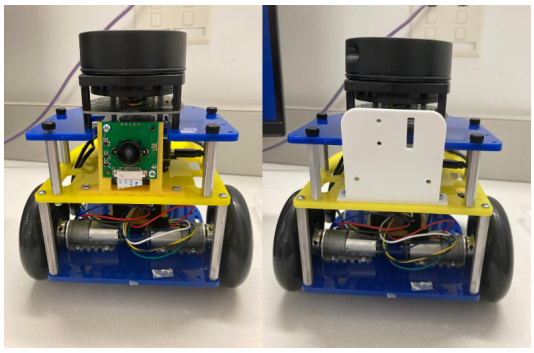

Figure 2: Mbot with the standard camera mount (left) an the RescueBot with our custom camera mount (right)

Generating a heatmap of the RescueBot's surroundings required the fusion of sensor data between the IR Thermal camera and the intel realsense depth camera. As shown in Figure 3, the MLX90640 IR Thermal Camera is able to output a 32x24 pixel image, with each pixel representing its temperature in celsius. This means that in each frame, we are able to capture 768 temperature points. As shown in Figure 4, the D435i camera outputs a 640x480 pixel image, with each pixel representing the depth (distance away from the realsense camera) of an object. Therefore, through utilizing these two cameras ,we are able to obtain the depth as well as the temperature information of any pixel.

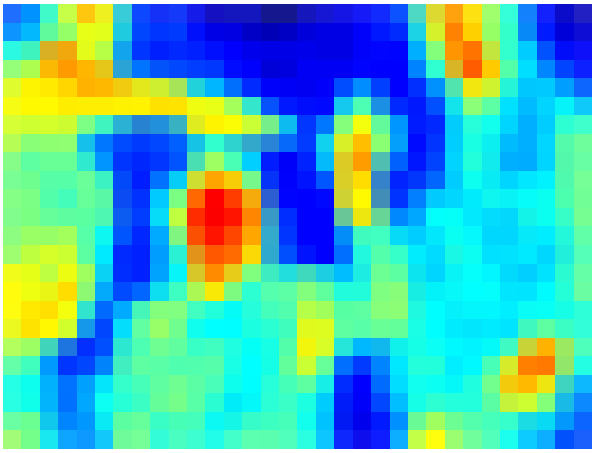

Figure 3: Live Output image of MLX 90640 thermal camera (upscaled to 640x480 pixels)

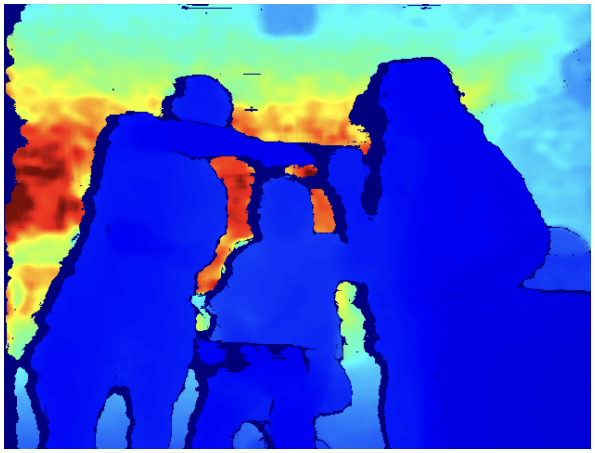

Figure 4: Live output image of intel realsense D435i depth camera.

The Intel Realsense camera and the thermal camera both have different resolutions, lens distortion, field of views, and even orientation and perspective. Therefore, it was difficult to correlate each pixel of the realsense depth camera to each pixel of the thermal camera. This is evident when we compare the output images in Figure 3 and Figure 4. We can see that the realsense depth camera has a much wider field of view than the Thermal camera. In addition, the perspectives of these two images are vastly different due to the intel realsense camera being fixed slightly below the thermal camera. As a result, we can clearly see that the positions of objects in the thermal camera vary greatly from the positions of the same objects in the realsense camera.

Initially, we experimented with an approximate linear interpolation of each pixel in the thermal camera's

frame to the Intel Realsense depth camera's frame. This approach is illustrated below in Figure 5, where each pixel coordinate in the thermal frame is multiplied by the resolution ratio of the depth and thermal frames and added to an offset demarking the origin of the smaller thermal image in the larger depth image.

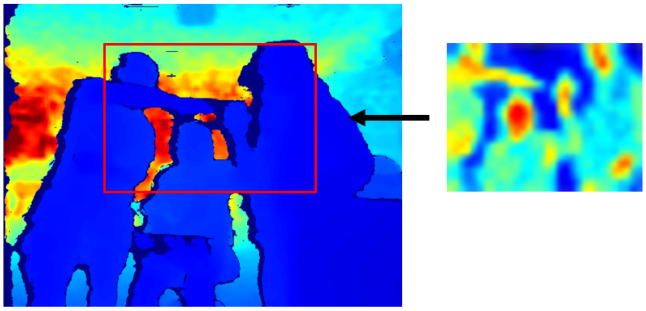

Figure 5: Linear interpolation of pixels between frames

However, we quickly found through testing that this method has large errors as it is difficult to locate the same points in both images accurately using the human eye alone.

Instead, we utilized homography, a process depicted in Figure 6, to relate the pixels in the thermal frame to the pixels in the depth frame. We were able to do this as we assumed the pinhole camera model and that both the thermal camera and the realsense depth camera captures an image in the same planar surface. Therefore, we were able to find a homography matrix that was able to shift the perspective of the image produced by the intel realsense depth camera to the perspective of the thermal camera. This was done by multiplying the pixels in depth image by the Homography matrix using OpenCV.

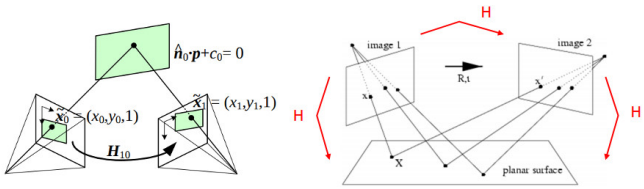

Figure 6: a Homography matrix H is used to relate multiple camera views of a given scene

To calculate this homography matrix, we first needed to find multiple matching keypoints between the images of the depth camera and thermal camera. For RGB cameras this is typically done with a checkerboard pattern. However, this approach does not work with thermal or depth cameras as the checkerboard features do not appear in either of them as there is no variation in thermal radiation of depth across the checkerboard.

Instead, we created our own feature matching process. We first captured the same frame with both the depth camera and thermal camera. To match keypoints, we utilized 6 tea lights to generate heat points in the thermal camera. We also raised these tea lights on bottles so that we were able to capture the depth of these 6 tea lights. Note that this calibration method would not work if we simply set the tea lights on the floor, since the depth camera would not be able to distinguish between the floor and the tea lights as they are at the same elevation. This calibration method is shown in Figure 7.

Figure 7: Our camera calibration environment consisting of tea lights, producing heat for the thermal camera, on top of plastic bottles, producing changes in depth for the depth camera

After capturing the frame of the tea lights with both cameras, we were able to clearly identify pixels in the thermal image that corresponded to pixels in the depth image. This is evident in Figure 8. The left picture is the thermal image and the right picture is the depth image. Each matching keypoint is identified by the green line. These matching keypoints were then fed into OpenCV's findHomography function, which allowed us to find the following Homography matrix:

$$\mathrm{H} = \begin{bmatrix} 1.74135451e+00 & 1.01058261e-01 & -3.23461390e+02 \\ -1.73114838e-01 & 1.90259195e+00 & -1.23924668e+02 \\ -3.63572321e-04 & 6.43620375e-05 & 1.06056044e+00 \end{bmatrix}$$

Equation 1: Homography matrix used for depth and thermal camera fusion

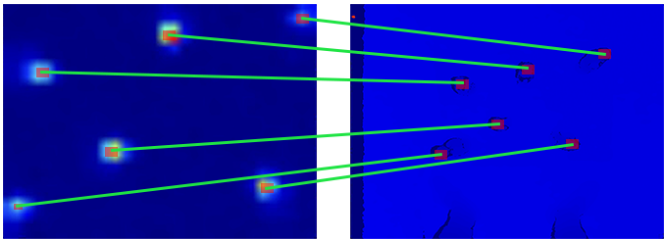

Figure 8: Pixel correspondence for the thermal camera output (left) and the depth camera output (right)

Given that the distance away of an object from RescueBot is known from the Realsense depth camera, the horizontal distance of the object with respect to the robot can be calculated using the pinhole camera model and thermal camera focal length. The equation for determining the horizontal distance X is shown in the Equation 2 below, where f is the focal length, x_{pixels} is the number of pixels away from the center of the image, and Z is the known distance of the object away from the camera.

$$x_{pixels} = \frac{fX}{Z}$$

Equation 2: Determination of horizontal distance using camera intrinsics and depth reading

We determined the focal length of the thermal camera experimentally. A lit candle was placed at a known horizontal distance away from the forward axis of the camera. The camera was then moved to different depth positions away from the candle, where the horizontal pixel offset of the flame in the thermal image was measured. The experiment setup can be seen in Figure 9 below.

Figure 9: Experimental setup to determine the thermal camera focal length. Overhead view of RescueBot and a tea light offset from the RescueBot's center (left) and the corresponding thermal output (right)

From our experiment data, we determined that the focal length of the thermal camera was approximately $f = 38.1$. The full calibration data can be seen in the chart if Figure 10 below.

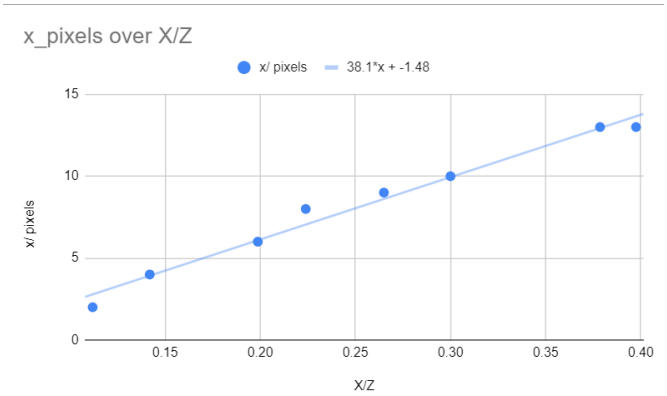

Figure 10: Resulting calibration data of manual determination of thermal camera focal length

The depth camera returns the (x,y) pose of thermal objects in the robot's local coordinate frame. In order to find the world position of these cells relative to the starting position of the robot, a linear transformation must be applied to these poses. The differing coordinate frames these poses relate to are illustrated in Figure 11.

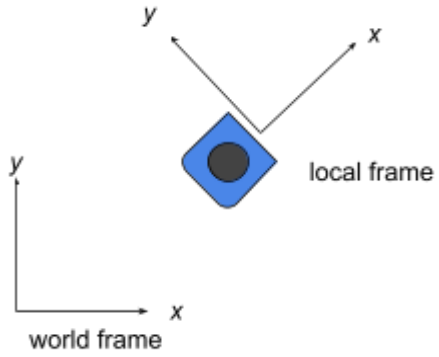

Figure 11: The RescueBot operates between the local and world frame, both requiring different descriptions of the same pose

The (x,y,θ) pose of RescueBot is determined continuously using the SLAM model implemented in Botlab. The world position of thermal objects can be calculated with the following calculations:

$$x_{world} = cos(\theta)x_{local} - sin(\theta)y_{local} + x_{pose}$$
$$y_{world} = sin(\theta)x_{local} + cos(\theta)y_{local} + y_{pose}$$

Equation 3: Conversion from the RescueBot's local coordinate from to the global coordinate frame

To create the thermal mapping, the RescueBot executes navigation of its environment using the A* algorithm and SLAM. The SLAM algorithm already generates an obstacle map based off of the Mbot's current estimated position and the LIDAR readings. This mapping was modified to have the heat-mapping overlayed, allowing rescuers to have an idea of both the physical and thermal layout of the environment. The Lidar sensor provides 360° coverage of the RescueBot, but the cameras required for heat mapping are mounted on the front of the RescueBot. To avoid heat sources going unrecorded during the mapping process due to unideal camera position, every five waypoints reached the RescueBot would complete a slow full 360° rotation to take in its surroundings.

Once the pairs of temperature and pose data are received, the location of the temperature reading relative to the Mbot's current position is calculated and the thermal map is accordingly updated.

Additionally, the IR camera has a fairly low resolution of 32x24 pixels, meaning each pixel can represent a significant area, leading to inaccurate generalizations of temperature conditions. To account for these issues, the mapping algorithm will increase or decrease the likelihood that the recorded temperature is accurate by considering the number of readings that place a given location at that temperature. The implementation of error correction using multi-pose measurements caused erroneous measurements fading away as more readings were recorded and accurate measurements shown more intensely, resulting in considerably improved final mapping.

The second task was to conduct search and rescue, in which RescueBot explores an unknown space until a heat signature of a person is found. It then attempts to lead the target to an exit using a safe path, away from sources of fire. We modified the exploration mode to include a -s flag signaling the RescueBot to be run in search and rescue mode.

In order to efficiently lead the survivor back to the home pose, we configured the motion controller to drive in reverse once it detects a survivor, which was simulated by a second Mbot with a tea candle. We used LCM messages to continuously inform RescueBot of the presence of a survivor by evaluating the thermal and depth data we received from the cameras. Once a heat source is detected within a predetermined range, RescueBot completes exploration and immediately begins returning home while driving in reverse.

While executing rescue, RescueBot continues to keep the target within a range of itself. Since the cameras are mounted on the front of RescueBot and have a limited field of view, it is possible that RescueBot will not detect the target if it remains still while waiting to detect the target. We created a waiting state to combat this problem. When the target is outside of this range, RescueBot enters the waiting state where it stalls until the target is detected within the range. We configured the bot so it will also spin slowly in place to detect any nearby heat sources from all angles. This enables RescueBot to spot the survivor once it is back in range.

Results

The accuracy of the heat mapping capability was evaluated by comparing the actual location of a set heat source from RescueBot's starting location to the recorded location of the heat source in the mapping. The mapping will be evaluated on two criteria: accuracy and safety. The accuracy of the mapping was determined by the difference between the mapped and recorded temperatures and the safety metric was determined by what percentage of free space that the robot considers to be safe is actually safe.

To test our fire detection model, we created a maze for RescueBot to explore. Tea lights were placed within the maze as heat sources. RescueBot explored the maze autonomously and generated a heat map representing the location of these tea lights. RescueBot marked all areas containing tea lights as dangerous using color markers on the BotGUI, and correctly labeled free space surrounding the fire with minimum error, as shown in Figure 12, accounting for noise discussed in the Methodology section.

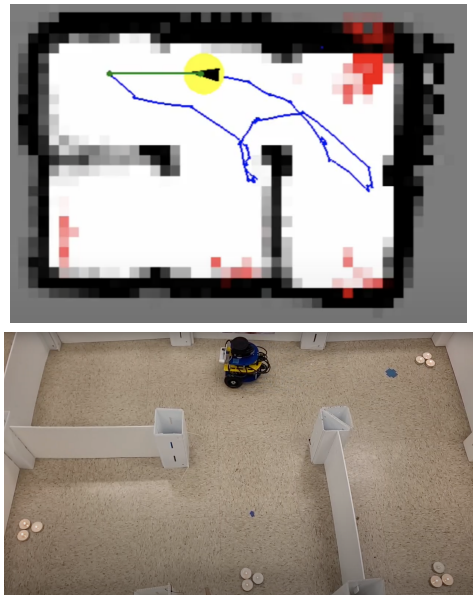

Figure 12: The fire detection map and path generated by RescueBot (top) and the actual map it was placed in (bottom). The generated map closely resembles that of the actual locations of the tea lights.

In addition, we evaluated the search and rescue capabilities of the RescueBot by whether it could efficiently plan a path from the survivor's current location to a safe location. We tested this by creating a maze similar to the maze we used to test fire detection, where a human target was simulated by another Mbot with a tea candle mounted on top of it as seen in Figure 13. The second Mbot was controlled via tele-op to follow RescueBot out of the maze. Upon multiple iterations, RescueBot was able to explore the maze until it detected the second Mbot, and led it to the exit or home pose of the maze, as shown in our video demonstration.

Figure 13: Teleop-controlled Mbot with mounted heat source used to simulated a survivor in need of rescue

Efficiency was determined by the speed of leading, shortness of path, and closeness of the final position to the determined safe location (the starting pose). The RescueBot was able to take a sufficiently direct path to safety by means of the A* algorithm and returned to the predetermined safe location within 10cm. The speed at which the RescueBot was able to lead the other Mbot to safety was limited by the need to keep that Mbot within the 0.5m minimum distance. Moving too quickly may introduce too much noise that could negatively affect the rescue process. As a result, the slow movement from pose to pose ensures accurate path leading but leaves room for optimization, as search and rescue situations require high speed movement.

Limitations and Sources of Error

Below are several sources of error in our thermal detection and localization model.

First, rapid rotational and forward motion can cause the sensor readings to be more inaccurate than if RescueBot was at rest as there is latency between the processing of the camera images and the motion of RescueBot. This is similar to the issues with LiDAR addressed with the MovingRayScan function, however the issue is even worse as the update rates of both cameras is only 4 Hz due to the limited processing power of the Raspberry Pi. In addition, there is a delay between the depth frame and thermal frame captured by the 2 cameras.

Secondly, we observed that open flames often have heat points above their physical bodies, such as the tip of the flame above a candle. The depth camera then measures past the candle and to the object behind, making it appear the thermal object is further from RescueBot than in reality. This can be seen in Figure 12 by the red heat points within the walls of the maze.

The third error is the low precision of both the thermal camera and thermal map. As each square in the thermal grid is 5 cm^2, multiple pixels in the thermal frame can correspond to the same point, leading to values being overwritten or modified as each frame is processed.

Lastly, we observed our homography transformation appears be less accurate for distances greater than 1.5 meters

Source Code and Demo

Demo video: https://youtu.be/Hb0IaKkzRS4
Code Repository: https://github.com/jastinjn/RescueBot

Discussion and Future Work

In the future, we propose adding several more features to the bot, the foremost of which would be to integrate the two modes: mapping and search-and-rescue, that currently operate independently on our bot. This would enable the bot to both map obstacles and fire hazards in its surroundings and also lead a survivor to safety at the same time. A challenge in adding this feature would be to first differentiate between the thermal signatures of a "human" survivor and that of the tea lights that we are currently using to simulate both fire hazards and survivors. This feature presents additional challenges in that the fires detected in the heat mapping portion will have to be treated as solid obstacles that the bot must effectively navigate around, since in our current algorithm they are only visible on the thermal grid.

If given more time, we would also consider adding light and audio instructions to RescueBot's search and rescue task. This would enable a human survivor to better locate and follow the Mbot in a dark setting. The Mbot will have flashing lights and output voice commands through a speaker. We will also add sound recognition, where human sounds detected on a mic will be prioritized and the bot will plan a path to the source of the sound.

Some of the technical challenges encountered during this project could be improved by using a more powerful computing unit. Blob detection using image processing could be used to locate objects in the thermal frame instead of naively processing each pixel in the frame as is done in our current algorithm. In addition, faster computing should reduce latency between the cameras leading to more accurate pose estimation of thermal objects

References

[1] "FDNY Buys 2 robot dogs to help with search and rescue operations," FireRescue1, 18-Mar-2022. [Online]. Available: https://www.firerescue1.com/fire-products/technology/articles/fdny-buys-2-robot-dogs-to-help-with-search-and-rescue-operations-mIe8HcwwFbdAB0I5/. [Accessed: 16-Mar-2023].

[2] "These search and Rescue Robots could save your life," Freethink, 08-Jul-2021. [Online]. Available: https://www.freethink.com/series/uprising/search-and-rescue-robots. [Accessed: 16-Mar-2023].

[3] Ackerman, E. (2022, August 18). *Robot scout finds fires with 3D thermal imaging*. IEEE Spectrum. Retrieved April 18, 2023, from https://spectrum.ieee.org/ucsd-firefighting-robot-3d-thermal-imaging