Deployment of Mobile Robot Networks Using Coverage Control

Matthew Wen
University of California, San Diego
Fall 2014

Summary

This report documents my preliminary work regarding deployment of the Turtlebots using coverage control. The overall goal is to distribute the robots across an area according to an 'importance' function. Background research of coverage control and relevant C++ techniques were conducted. An algorithm has been developed in MATLAB to generate importance functions based upon a Gaussian distribution. The algorithm has yet to be implemented into the current deployment code.

Big Picture

The purpose of the ROS Turtlebot project is to study and implement control algorithms to govern the behavior of robots within a multi-robot network. The overall goal is to take advantage of multiple robots and have them cooperatively complete tasks in the most efficient manner possible. Current studies include the use of SLAM algorithms to map out an area using multiple robots and the deployment of robots into specific arrangements or positions.

My Contributions

My work this past quarter specifically regarded the implementation of coverage control over multiple robots in order to achieve specific distributions of the robots over an area. Rokus laid the groundwork for implementing coverage control using two Turtlebots over an area defined by an overhead camera. His work culminated with a program that uniformly distributes two Turtlebots over an area defined by an overhead camera. My work builds upon the program that Rokus wrote. My specific goal is to develop a methodology to implement an 'importance' function or 'density' function across the area the Turtlebots inhabit, such that the Turtlebots distribute themselves based upon a given 'importance' function, rather than a uniform distribution.

My work is related to the work of Daniel and Katherine. Katherine was able to apply Rokus's code to situations involving more than two Turtlebots. Daniel is working on developing a model to prioritize regions of an area for mapping that have not been explored by a Turtlebot. The 'importance' function could be applied to this model by placing more weight on the unexplored regions of the area.

Preliminaries

For the Turtlebots, the process of coverage control occurs in roughly four steps. First, the Turtlebots must be localized on the area covered by the overhead camera to attain each Turtlebots' position. Next, Voronoi diagrams are generated which divide the total area into cells in which every point in each cell is closest to the Turtlebot that occupies that cell. Third, the centroid of each Voronoi cell is calculated and set as a goal point. Fourth, the Turtlebot moves toward the goal point.

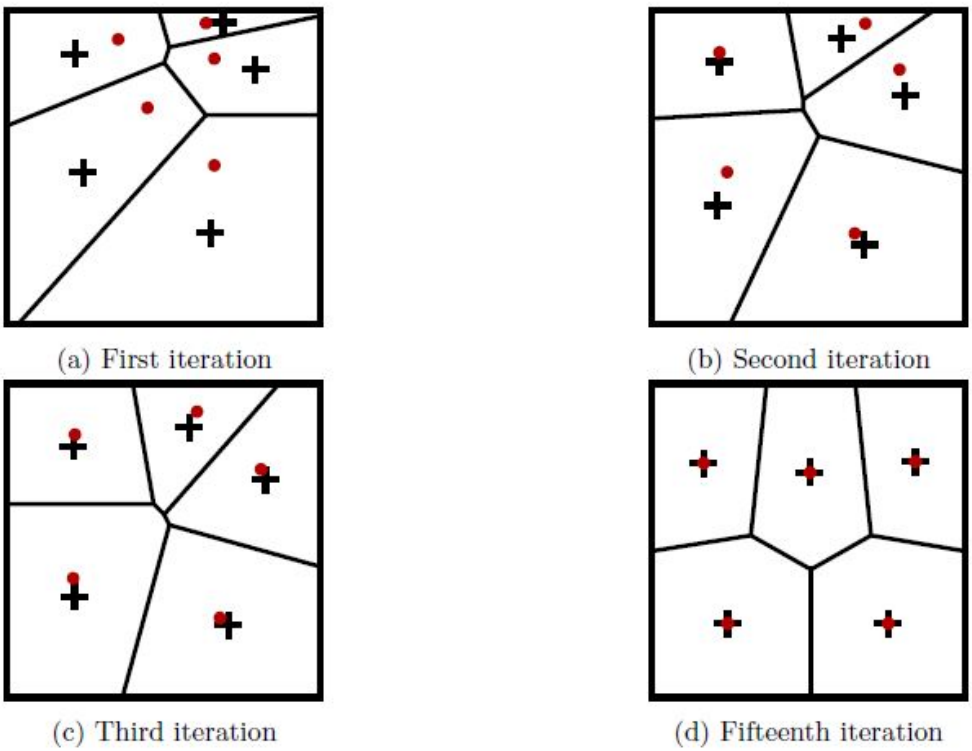

Figure 1 Successive iterations of deployment involving Voronoi cell diagrams. [1]

Methodology

In order to apply an 'importance' function that contains gradients and non-uniform weighting, gray scale imaging was chosen to represent the 'importance' function across the area the robots inhabit. In its current state, the deployment code uses black and white to define the boundaries of each Voronoi cell and split the Voronoi cells up into separate images. As a result, the image of each Voronoi cell is made up of an array of zeros and ones, with ones indicating that a pixel is part of the Voronoi cell and zeros indicating that pixels that are not part of the Voronoi cell. The current deployment program uses the array of each Voronoi cell to calculate the centroid. However, in order to apply a non-uniform 'importance' function, a gray scale image, in which each pixel has a value between 0 and 255, that represents the importance function of the area, is multiplied element-by-element with the image of each Voronoi cell. The result is a Voronoi cell with a gray-scale non-uniform 'importance' or 'density' distribution. Therefore, when the centroid of each Voronoi is calculated, the gray-scale, non-uniform distribution is taken into account, and a new centroid is calculated based upon the 'importance' function.

When this modification is applied across the network of Turtlebots, the final result is a non-uniform distribution of the Turtlebots with more Turtlebots positioned in regions of higher weight or importance.

I have written a MATLAB code that simulates a Gaussian importance distribution over the pixels of a 640x480 image, which is the image size generated by the overhead camera. The Gaussian importance distribution places a value between 0 and 255 for each pixel in the image. The next step would be to convert the MATLAB code into C++ to be implemented in the existing simple_deployment.cpp code by

multiplying the gray-scale image with all Voronoi cell images generated in the simple_deployment.cpp code.

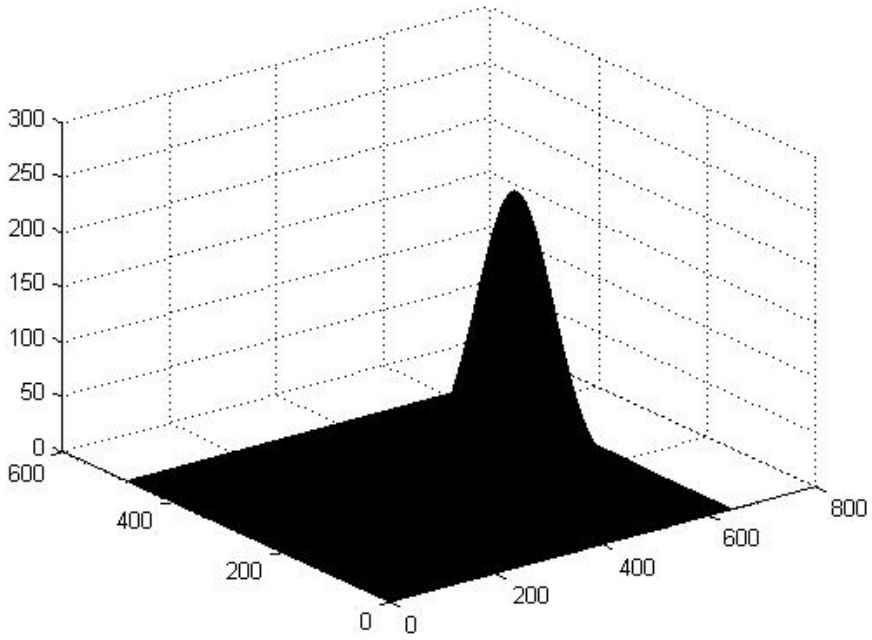

Figure 2 Gaussian importance function generated in MATLAB for a 640x480 size image.

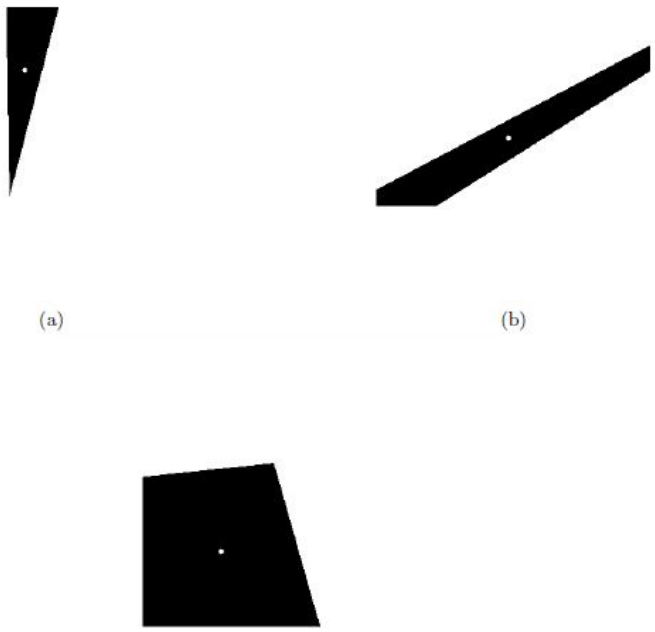

Figure 3 Voronoi cell diagrams images represented by black and white pixels. [2]

Tips

One issue I encountered was associated with ssh-ing into the robots. Sometimes the master computer would not recognize the presence of the Turtlebots on the UCSD-Protected network. Eventually, I discovered that some of the Turtlebots use expired login information or randomly reset their Wi-Fi connection so they need to be manually reset to the UCSD Protected network.

Another issue I discovered was that the camera node would not run properly while I was logged onto my account. However, the camera node works properly when logged into Katherine's account. This is due to a directory issue in ROS because the uvc_camera file that is called when the localization_demo.launch is launched is located in Katherine's profile on the Calipso computer. It is a temporary issue, but for the time being, deployment can only be run through Katherine's profile.

Pitfalls

As of this moment, I have not encountered any specific challenges that could not be identified and remedied. Most of my time this quarter has been spent learning ROS and C++, so I have not encountered complex issues that could not be resolved by asking another lab member.

Conclusions and Future Work

This quarter progress has been made on Rokus's code in that it can be applied to situations involving more than two robots. The 'importance' function aspect has been developed in MATLAB and should be implemented into Rokus's code within the next few weeks.

Future work regarding the deployment of the Turtlebots could involve having a dynamic environment instead of a static one for the Turtlebots to operate in. For example, while multiple Turtlebots are working toward distributing themselves about an area, one Turtlebot could be manually controlled and move about the area, forcing the other Turtlebots to react to this one Turtlebot's positioning. In addition, a dynamic 'importance' function can be introduced into the environment where the 'importance' function can change depending on a user input.

A more long-term goal would be to somehow apply the 'importance' function methodology to real-world scenarios such as prioritizing certain regions to explore for mapping.

References

[1] Dominik Moritz. Lloyd's algorithm. http://en.wikipedia.org/wiki/Lloyd's algorithm, 2013. Accessed: September 30, 2014.

[2] Ottervanger, Rokus. "Implementation of a Distributed Algorithm for Coverage Control." 2014.