Relative Localization for Mobile Robots using Image Processing

Jeremy Kanovsky

Abstract

A robust and high-speed implementation of localization and control of a swarm of mobile robots is described in this paper. The main purpose of this work is to verify the feasibility and reliability of a system relying on camera based processing for relative localization of multiple robots simultaneously. This approach enables deployment of multiple mobile robots in an unmapped and unknown environment. The proposed approach has been designed to run on low cost, low power robots for open-ended applications. Their performance of the presented approach is verified and validated in incomplete information simulations.

1 Introduction

A common problem in mobile robot swarms is localization of each robot relative to each other. The problem of absolute position for a single, or even multiple robots, has been solved in numerous ways, but relative localization is a relatively unexplored field. This paper will be exploring methods for developing an image based system for localizing a swarm of robots in an arbitrary environment with each robot aware of its own position and the positions of those around it. The robots will start with no prior knowledge of their own position, or of their absolutely position in the environment, which is also unknown and unmapped.

This paper also addresses a number of smaller problems along the way as the system is developed. Real-time communication between mobile robots will be tackled, along with real-time image process on extremely low power processors. The work done on this paper is targeted at running on low cost, low power robots with the theory of being application on all robotic platform that operate primarily in a plane (i.e., ground based mobile robots).

The goal of this paper is to explain how a simulation is developed such that we can validate the system works, and to replicate the interface a robot might have with hardware. In doing so, it can, in theory, be directly applied to robotic hardware. This paper will also cover the means and methods by which a simulation was created to provide each robot with limited information about itself and it's surroundings, similar to real-world environments.

2 Background & Related Work

Recent research of multi-robot systems has focused on high speed, high reliability techniques for relative localization using a variety of techniques (Kai Lingemann et al. 2005 and Thrun et al. 1998). An extensive amount of research has accumulated in the probabilistic field for locating, localizing, and reacting to other bots within a system (L. Jetto et al. 1999, A. Howard et al. 2002, and C. F. Olson 2000). While some rely on more advanced sensors and higher levels of processing, research has begun to move it's attention to lower cost vision based systems of localization (Giovanni et al. 2001 and Saska et al. 2017).

Unfortunately, most of these systems still rely on unrealistic system constraints and rely on ideal lab settings for their results. Alternatively, other systems include real-world conditions but do so on robots that would not function without high-bandwidth internet capability, or computer server processing capacity. Pose detection in real-time of systems has been solved for systems with absolute positioning (i.e., methods of localizing a robot relative to the entire environment and providing precise positioning coordinates), but less work has been done toward relatively localizing (i.e., giving coordinates relative to each other robot, but without prior or accumulated knowledge about the environment in which the robots exist). This paper will focus on the relative localization of robots, on systems with extremely limited processing and bandwidth capability.

3 Technical Approach & Methodology

There are a number of aspects to this project that must be addressed to make it viable. As mentioned above, the simulation being developed will target a mobile robot base. We will assert that each robotic base will be equipped with a QR code encoding information about which robot is which, and which face is currently being seen by a camera. We will also assume that the operator has access to the ground truth of the current state of each robot (for purposes of testing and debugging. This is also a reasonable assumption because in real-world lab settings external monitoring devices, such as VICON[1] cameras, might be used to achieve such a results). We will also assume that each robot is equipped with a camera, and axial encoders to measure the translational motion of the drive base. We will also assume these robots have internet capacity, and are capable of connecting to a wireless network.

In order to enable high-speed reliable communication between the robots, a custom wrapper for TCP sockets will be used. This library was developed specifically with robots with limited processing capability and their hardware limitations in mind. Each robot will have a web-socket connection that is fully connected to each other robot in the swarm. This system has inherent fault prevention in that the individual connections between robots are far less important than the graph as a whole, and nodes (robots) might create or lose an

[1] `https://www.vicon.com/`

arbitrary number of edges (socket connections) during the course of trials or deployments. Source code for this library can be found here with implementations in Python and Node.js: `https://github.com/0xJeremy/socket.engine`. This socket network will allow robots to report information that they see to the swarm as a whole (more information below in the Experimental Results section).

Each robot will maintain a matrix of it's own position (assuming itself to be point 0, 0 in an x-y coordinate plane), with the relational data of every other robot in the system. When another robot is encountered by the camera, the information encoded on the theoretical QR code of the other robot will be decoded, and the information about the location of the robot will be updated in the estimation matrix (or added, in the case the robot has not been seen before. This system assumes each robot has a globally unique identification number, or GUID). When the relevant information has been calculated, the robot will report it's findings to the swarm as a whole.

As part of the update step in the estimation matrix, each robot will maintain an estimate of it's own position based on it's translational movement data. We can assume the robot will have access to such data given the assumption of rotary encoders being present on the translation motors of the robot. This will allow it to also update the estimated locations of the other robots in the swarm as it moves, and keep them relative to itself (assuming they have not moved since last observation. More information about estimate updating in the Experimental Results section).

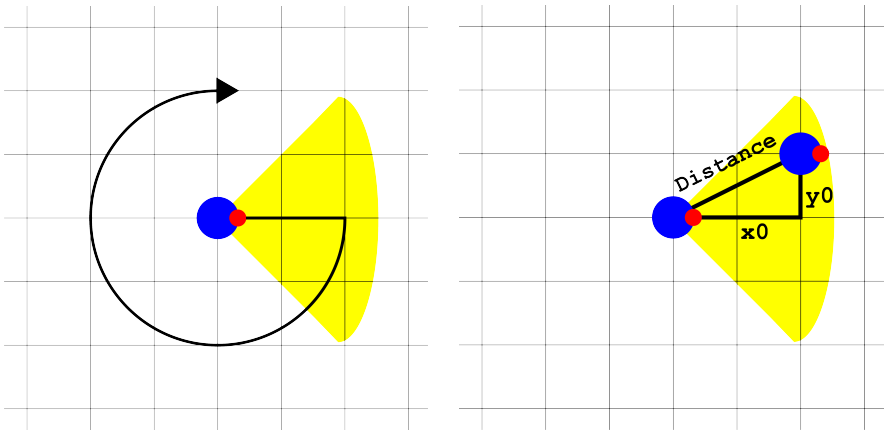
This simulation will also provide a ground truth state for the operator to observe and control. Because we are assuming a lab setting in which these robots will be tested, we can use the information from a ground truth state to calculate any error in the system by finding the Euclidean distance between each robot estimation and the actual position of each robot. This error metric will be used during development of the positional estimation algorithm, and compare the accuracy of each model over time to ensure regression does not occur in the software. This metric will also be used to ensure localization drift does not occur, or is minimized, over time.

3.1 Notation and Problem Formulation

In Figure 1 we can see the field of view of the robots in simulation. Each robot has a predefined distance it can see (as setup in the simulation parameters), and a maximum angle from the center (together these define a cone of view from the camera). If another robot exists within this cone, it is visible to the camera of the first robot. Each robot also maintains an angle as part of its pose. This pose is zeroed facing right, with an increase in angle going clockwise (see Figure 1a).

When a robot captures another in it's field of view, the distance from the robot is given, along with the angle θ from the center of the field of view. Using these two metrics, we can calculate the displacement in the x- and y-direction:

$$x_0 = cos(\theta) * distance$$



(a) Increasing robot angle (b) Displacement Calculation

Figure 1: Robot Field of View in Simulation

$$y_0 = sin(\theta) * distance$$

The formulas above are those performed on each robot when it observes another. They are also done in reverse by the simulation to determine where a robot should be reported, after it has been verified to be in the field of view of the robot. Below is the code implementation to determine if a robot is visible, and if so, at what angle (these are both done by the ground truth of the simulator):

```
def angle(ref, other):
  diff_x = other.x - ref.x
  diff_y = other.y - ref.y
  if diff_y == 0:
    return 0 if diff_x > 0 else 180
  if diff_x == 0:
    return 90 if diff_y > 0 else 270
  if diff_x > 0:
    if diff_y > 0:
      return degrees(atan(diff_y / diff_x))
    return degrees(atan(diff_y / diff_x)) + 360
  return degrees(atan(diff_y / diff_x)) + 180

def visible(ref, other, max_distance, max_angle):
  d = distance(ref, other)
  if d > max_distance:
        return False
  a = angle(ref, other)
  diff_angle = a - ref.angle
  if abs(diff_angle) <= max_angle:
```

```
    return True
return False
```

4 Experimental Results / Technical Demonstration

The experiments performed on this system were done with some constant variables across each trial. Instead of generating a series of random commands for each robot to follow, they were controller by an operator. This allowed experimental validation at every increment in the simulation.

4.1 Experimental Setup

The setup for the simulation was thus: a map of predetermined size was generated with a number of robots randomly placed around the simulation (the number being a pre-determined constant). These robots are entirely unaware of their own position, or the positions of other robots. They are unaware of their own pose angle. This simulation was decided to be two dimensional to simulate mobile robots with a driving capacity in a plane (see further work for more details about rough terrain environments).

The operator may take control of any of the robots at any time. They have the capacity to move the robot forward or backward (similar to the motion available to a two wheeled robot with a "tank drive" wheel configuration). The operator may also rotate the robot in place around it's center axis. These moves were determined the be readily available to real world systems. The operator might also command the robot to take a picture. In doing so, the simulation would project a potentially available view from the robots camera and calculate which robots are visible to the one taking the picture. This data would then be reported back to said robot in a similar format as if a camera had taken the picture and spotted a QR code on another robot. Information about the position of the robot was not given, only the size and displacement from a center line in the cameras field of view was provided to the robot.

The operator might also have the robot report it's information to the rest of the swarm. In doing so, the robot would encode all information it has about itself and other robots it may see and send the information across a network data socket to the other robots. All robots receive the same set of information, regardless of if they are in the field of view of the reporting robot. Lastly, the operator has the capability to have each robot read from the network socket and perform localization based on the new information. When a robot receives the information from another, it performs it's own calculations to determine the relative position estimate of the other robot to encode in it's own position matrix.

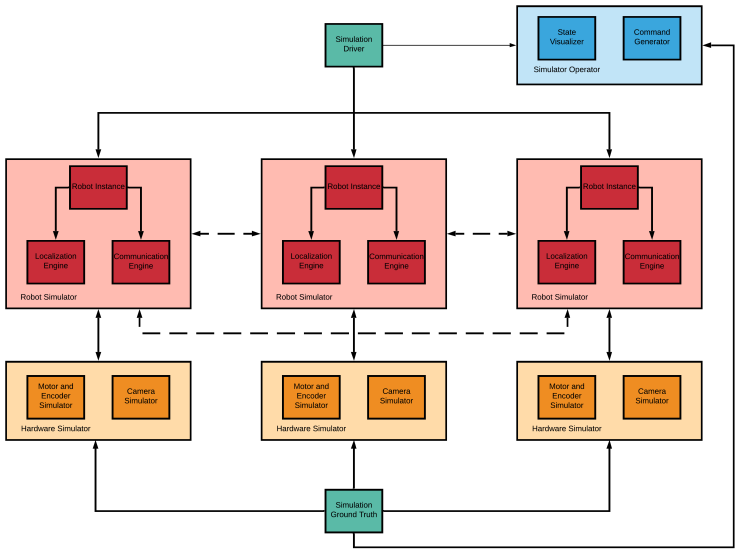


Figure 2: Software Implementation of the Localization Simulation

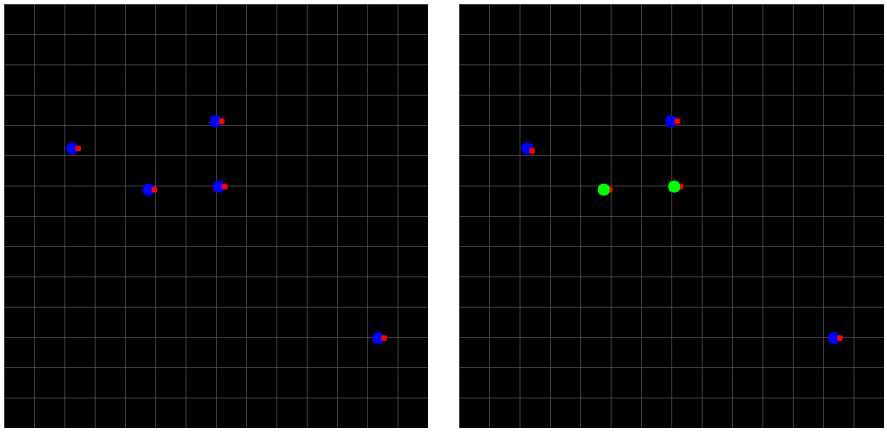
4.2 System Validation

To validate the system described above, trials were run manually to determine the accuracy and correctness of the mathematical calculations performed in localizing other robots, or updating ones own estimates. After the simulation was started, the operator would take control of a single robot. Because of the nature of the system, validation of a single bot would also validate the system as a whole, given that all robots rely on the same calculations. This robot was then oriented such that it had several other robots in it's field of view, some being within the "view" distance of the robot, and others falling outside the parameters of what the camera simulator would report as visible. This robot would then take a "picture", and calculate the position of all other robots it could see. In doing so, the simulation would mark these robots with a green dot. The Euclidean distance between the estimate and the actual position can be calculated, and was found to be within +/- 1 simulation unit for simulations without added noise (a simulation unit being an arbitrary measure of distance, and shares only a proportional relationship to real-world systems).

The operator would then command the robot to serialize and encode it's findings, and share them over the network socket. Due to the asynchronous method of communication, all messages would be received and processed immediately, but not acted on by the other robots until the operator gives more instructions. Each robot would then be told to read from the network socket, and perform their own calculations. These included finding the original robot based on it's estimate for each. They would, in essence, be doing the back calculation to find the original robot. The original robot would be marked with a red

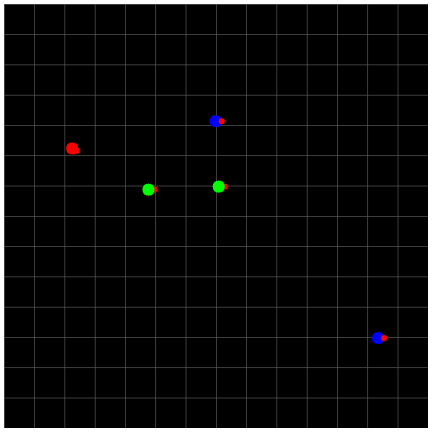
dot. Each robot would also have now encoded the estimated position for each other robot in the reported data. These estimates would remain until newer information is provided about each robot (for more details see the Future Work section). This method of back-calculation is the direct inverse for the original localization calculations, and therefore shares it's accuracy.

4.3 Experimental Results

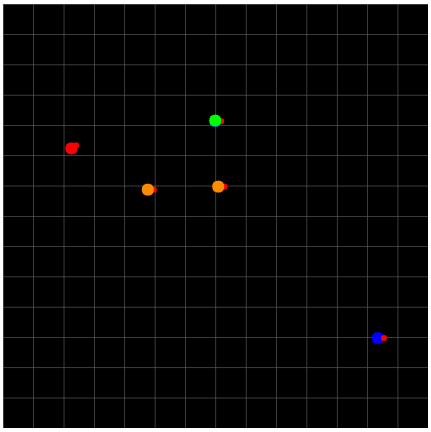


(a) Initial State

(b) Robots Localized



(c) Reverse Localization



(d) Second Localization

Figure 3: Simulation Results

The purpose of this simulation was to replicate the conditions of a hardware systems, and include the appropriate limitations. The software developed in the simulator could easily be applied to physical robots and was designed with that intention. The simulation was also meant to provide validation of the math sup-

porting the localization based on the image data available from a robot camera of the other robots in the system. Using this as an evaluation method, and using the error in the robot estimates (i.e., the Euclidean distance between actual robot and localization estimate), this simulation was a success. It validated the work supporting the theory, and showed that despite imposed hardware constraints the system would be capable of supporting the methodology.

5 Future Work

Despite this papers efforts to impose limitations upon the simulation similar to a real-world scenario, there are a number of things that might be improved to the make the system more robust. These begin with adding simulation noise. In the current system, the camera is capable of providing unobtainable good results and can provide robots with the exact distance and angle from center of each robot in it's field of view (and within the simulation parameter view distance). While remarkably good camera processing techniques exist for QR codes and April tags, there is still some fundamental error in distance measurements for monocular systems. To improve the simulation, it would be prudent to add error to each measurement making them less certain of the true state of other robots.

This leads into a second improvement. Currently, each robot will update it's localization matrix each time it receives new information about another robot. It will not, however, keep a record of past locations. It will also to continue to believe each robot will stay stationary until another estimate is provided or the robot is observed directly. It would therefore be beneficial to add a probabilistic estimate of the robot using multiple measurements when available in an appropriate time span, or by making estimates about it's movement when not visible. This system would benefit from a particle filter implemented to solve this constraint.

This leads to the estimation of each robot itself. As part of it's estimation process, each robot forms an estimate of where itself is in the world, assuming it is the origin of it's own plane. When the robot moves however, it receives perfect information from the simulation about how far it has moved, and what it's angular displacement was when turning. Methods exist for obtaining extremely accurate information as to a robots displacement given rotary encoders attached to the wheels, but they are still unlikely to be perfect. Adding a Kalman filter to run on each robot to provide estimates of it's own position would benefit the simulation by allowing noise to be introduced to the movement vector.

These improvements would enable the simulation to provide a more realistic approximation for the real world, and robots operating with actual (imperfect) hardware.

6 Conclusion

The goal of this work was to investigate a potential implementation for performing relative robot localization on low cost robots, and determining if such a system might be able to be deployed. The simulation developed for this paper shows that this implementation is effective in achieving this goal, and capable of localizing any number of robots in a swarm relative to one another. It also shows how estimates might be maintained for robots, even if they are no longer observed in the swarm.

There are a number of avenues in this research for future work, that, while outside the scope of this paper, would greatly improve the efficacy of the system if it were deployed to hardware. The software developed in this research can be deployed to actual robots for future work and testing.

References

1. A. Howard, M. J. Mataric and G. S. Sukhatme, "Putting the 'I' in 'team': an ego-centric approach to cooperative localization," 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422), Taipei, Taiwan, 2003, pp. 868-874 vol.1.

2. Saska, M., Baca, T., Thomas, J. et al. System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization. Auton Robot 41, 919–944 (2017). https://doi.org/10.1007/s10514-016-9567-z

3. Kai Lingemann, Andreas Nüchter, Joachim Hertzberg, Hartmut Surmann, High-speed laser localization for mobile robots, Robotics and Autonomous Systems, Volume 51, Issue 4, 2005, Pages 275-296, ISSN 0921-8890, https://doi.org/10.1016/j.robot.2005.02.004. (http://www.sciencedirect.com/science/article/pii/S0921889005000254)

4. Thrun, S., Burgard, W. & Fox, D. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. Autonomous Robots 5, 253–271 (1998). https://doi.org/10.1023/A:1008806205438

5. L. Jetto, S. Longhi and G. Venturini, "Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots," in IEEE Transactions on Robotics and Automation, vol. 15, no. 2, pp. 219-229, April 1999.

6. A. Howard, M. J. Matark and G. S. Sukhatme, "Localization for mobile robot teams using maximum likelihood estimation," IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, Switzerland, 2002, pp. 434-439 vol.1.

7. C. F. Olson, "Probabilistic self-localization for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 16, no. 1, pp. 55-66, Feb. 2000.

8. Y. S. Hidaka, A. I. Mourikis and S. I. Roumeliotis, "Optimal Formations for Cooperative Localization of Mobile Robots," Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 2005, pp. 4126-4131.

9. Giovanni Adorni, Stefano Cagnoni, Stefan Enderle, Gerhard K. Kraetzschmar, Monica Mordonini, Michael Plagge, Marcus Ritter, Stefan Sablatnög, Andreas Zell, Vision-based localization for mobile robots, Robotics and Autonomous Systems, Volume 36, Issues 2–3, 2001, Pages 103-119, ISSN 0921-8890, https://doi.org/10.1016/S0921-8890(01)00138-5. (http://www.sciencedirect.com/science/article/pii/S0921889001001385)