# EXPERIMENTAL TESTING OF AN AUTONOMOUS RADIATION MAPPING ROBOT

Michael E. Hosmar [1], Scott B. Nokleby [1], and Ed Waller[2]
[1]*Mechatronics and Robotic Systems Laboratory, University of Ontario Institute of Technology, Oshawa, Canada*
[2]*2 Faculty of Energy Systems & Nuclear Science, University of Ontario Institute of Technology, Oshawa, Canada*
*Email: michael.hosmar@uoit.net, scott.nokleby@uoit.ca, ed.waller@uoit.ca*

## ABSTRACT

Radiation maps provide an easy to understand view of the invisible hazards that radioactive sources pose. Previous methods of producing radiation maps either required prior knowledge of the physical dimensions of the mapping area or needed the intervention of a human operator. In this work, the experimental test results of the radiation mapping capabilities of a fully Autonomous Radiation Mapping Robot (ARMR) are presented. Two scenarios are constructed in an outdoor environment. The first test uses two Cs-137 sources with different intensities. The second test has one large source placed in the area. The results demonstrate the effectiveness of the robotic system.

**Keywords:** radiation mapping; mobile robotics; autonomous sampling.

## 1. INTRODUCTION

Radiation mapping, like other sensor mapping techniques, requires a source model and a method for determining where measurements should be taken (called exploration). The existing literature describes different methods for modelling and localizing radiation sources [1–3]. In each case, a sensor needs to be moved through the environment to the sensing locations in order to take each measurement. This movement can be accomplished with the assistance of a human operator or with previously affixed sensors. However, this either exposes a human to possibly harmful doses of radiation or requires that sensors be put in place in the environment to be mapped before the need for a radiation map arises. The cost of a large number of sensors and the need for preplanning makes this an unlikely situation during a disaster. In order to move radiation sensors safely and automatically through an environment, an autonomous mobile robot was designed and implemented.

There has been previous research into the use of mobile robots for the purpose of radiation mapping by Cortez, et al. [4] and McDougall, et al. [5]. Cortez, et al. implemented a fully autonomous algorithm for exploration and movement. However, this system relies on prior knowledge of the configuration of the space in order to plan a route through it. McDougall, et al. implemented a robust source localization system using a robot to take measurements. However, the robotic system was not completely autonomous and relied on the operator to guide it through the mapping area. Both systems lacked a chassis and localization system suitable for mapping outdoors.

The design that is proposed here builds upon the research of McDougall, et al [6]. McDougall, et al. proposed the use of Robot Operating System (ROS) and the corresponding open-source libraries. For this work, a more robust robotic base was selected for outdoor environments and a different robot localization strategy utilized to navigate outdoors. Additionally, an autonomous exploration algorithm was developed to move the robot through an unknown environment. A new source localization algorithm was also developed to better use the data from the exploration algorithm.

In this work, the effectiveness of the robotic exploration algorithm and source localization algorithm are investigated through real-world testing in an outdoor environment. The exploration algorithm decomposes the search area provided through a Graphical User Interface (GUI) into discrete planned measurement positions. Robotic localization is performed through traditional techniques utilizing inertial sensors, magnetometer, and a Real-Time Kinematic (RTK) GPS. The robot navigates through each point taking measurements automatically. At the end of the search the sources are localized using a Particle Swarm Optimization (PSO) algorithm [7]. The effectiveness of the system is evaluated by producing a radiation map of an outdoor environment with an unknown configuration of sources and comparing the results to surveyed locations.

## 2. SYSTEM OVERVIEW

The autonomous radiation mapping robot physically consists of a robotic base and external sensors. The robotic base is a Unmanned Ground Vehicle (UGV) manufactured by Clearpath Robotics called the Jackal (see Figure 1). Added to the outside of the Jackal is a LiDAR and a Na(I) (sodium-iodide) radiation detector as well as a RTK GPS rover node.

The Clearpath Robotics Jackal is a robust and compact mobile ground robot. The Jackal consists of a rectangular body with four wheels around the perimeter. Inside the body, is an electric battery power system and a full computer with WiFi and Bluetooth. The computer allows the Jackal to control its motors, process its sensors, and perform autonomous navigation and exploration on board.

Internally the Jackal has many sensors for navigation. The motors are connected to optical encoders. The encoders provide rotation measurements of the motors and through a fixed gear ratio they provide the rotation measurements of the wheels. Additionally, there is a Magnetic Angular Rate Gravitation (MARG) sensor for inertial attitude sensing of the robot on the Jackal's control board. All sensor output is published

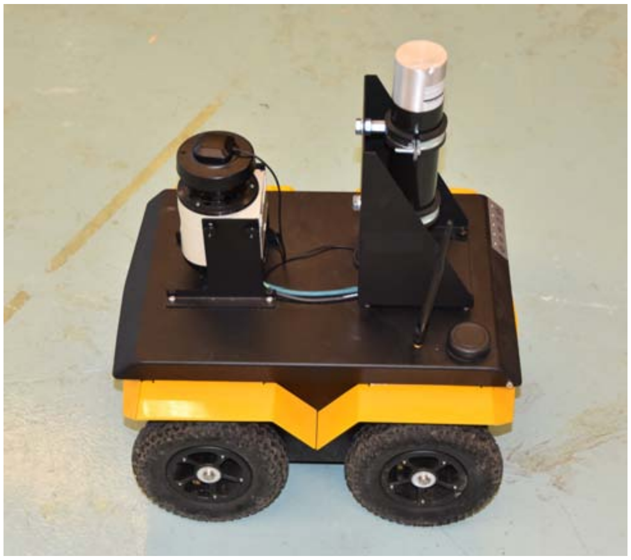


Fig. 1. The Jackal robotic platform by Clearpath Robotics.

to ROS interfaces through the on-board computer.

The radiation detector utilized is a Rexon Na(I) scintillation detector with a photomultiplier tube permanently affixed. The detectors signal is read by a URSA Radiation Alert DAC which is housed in the robot's case. The DAC is controlled and read by a custom ROS software node. The data is relayed to a ROS compatible interface.

## 3. GPS LOCALIZATION

In addition to the classical techniques of sensor fusion for pose estimation and localization, an RTK GPS was used to improve localization accuracy outdoors. In an outdoor environment there may not be a sufficient amount of landmarks for a Simultaneous Localization and Mapping (SLAM) algorithm to localize against. This would be the case where the robot traverses an open field. In these scenarios the robot will have to rely on the GPS and on board odometry to produce a location estimate.

During preliminary tests in a built up courtyard, it was noted that the GPS accuracy was greatly affected by the buildings surrounding the courtyard. The surrounding buildings blocked GPS signals from satellites near the horizon, which greatly affects the accuracy of the position estimate. The buildings also reflect other GPS signals which introduce multi-path errors into the localization estimate. After conducting two tests gathering position estimates from a stationary GPS receiver stationed within the courtyard and then an open parking lot the difference in accuracy can be seen. The move to a more open area improved the position estimate and reduced the stationary error from around $\pm 4\ m$ to $\pm 1\ m$. This may be an acceptable accuracy for most civilian needs. However, the accuracy of the results of the PSO algorithm and the heatmap will be affected by this drift. It is unlikely, due to the high variability of the cost function and its inflexibility to erroneous data, that the source localization algorithm would converge to an accurate position with this amount of static error.

It was decided to upgrade to a more accurate positioning system to increase reliability and accuracy, especially when traversing open terrain. To accomplish this, an RTK GPS system was used. An RTK system improves on a standard GPS system by leveraging two different techniques. The first technique analyses the carrier frequency of the GPS signals as well as the transmitted data. Since the carrier frequency is much higher than the data that it carries, the system can more accurately identify the apparent distance to each satellite. The second technique is differential GPS. Differential GPS is the process of applying double differencing techniques using two GPS receivers to effectively eliminate distortions caused in the atmosphere. The combination of both techniques is RTK GPS [8].

The GPS chosen for the RTK system in this work is a kit developed by Igor Vereninov (https://emlid.com/reach/). It consists of two inexpensive GPS units with a single board computer paired with each GPS to do the RTK processing. The RTK processing is accomplished with an open source software package known as RTKLIB [9]. This software was developed to perform any of the RTK GPS functions that a proprietary system could, provided that the user has compatible hardware. The system is capable of outputting National Marine Electronics Association (NMEA) data, which makes it possible to interface with the existing ROS nodes in the system.

Using this system the positioning accuracies were greatly improved. The achieved accuracy could be as low as $\pm 0.05$ *m*, however, in practice it was around $\pm 0.15$ *m*, still a significant improvement. The RTK GPS added complexity to the physical implementation by adding the need for communication between a stationary base station and the robot and a known surveyed location to place the base station. The communication between the two GPS units was accomplished using the existing WiFi connection between the robot and the control station. Obtaining a surveyed location was accomplished using the base station GPS itself to survey its location. A GPS log was obtained over a period of around ~45 minutes and sent to Natural Resources Canada's Canadian Geodetic Survey for free processing into a highly accurate position. The method used by Natural Resources Canada, called precise point positioning, utilizes data collected from highly accurately surveyed base stations throughout Canada to model errors in GPS ephemeris and clock data. This method allows a single receiver to be used to find an accurate position.

## 4. AUTONOMOUS EXPLORATION AND SOURCE LOCALIZATION ALGORITHM

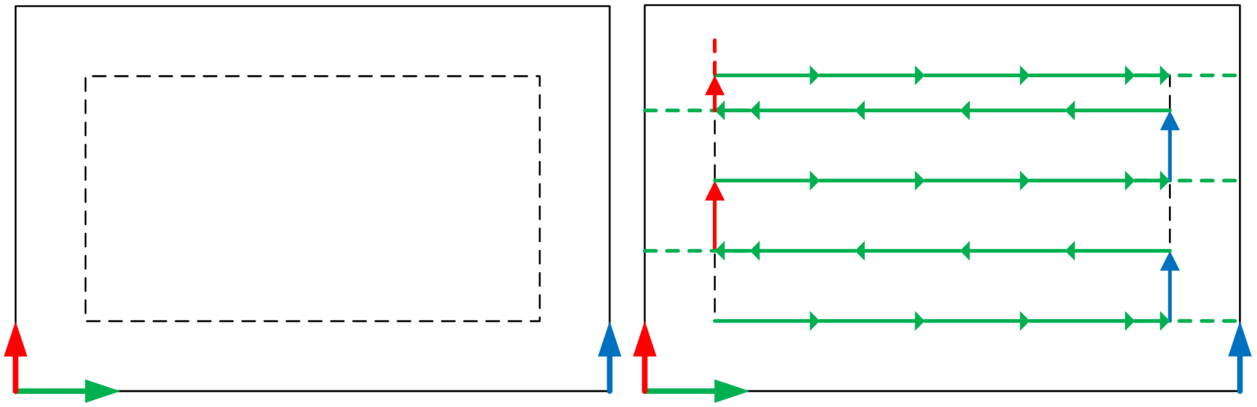
### 4.1. Autonomous Exploration

The exploration algorithm takes in a four-sided exploration polygon and produces a plan for exploration in the form of waypoints. These waypoints are used as instructions for where to take measurements and are handled by the navigation system. The generated plan needs to cover the entirety of the area of interest and do so evenly. The proposed method uses a variable grid to accomplish this.

The operator first decides on an appropriate row spacing and goal spacing. This could be based on the environment searched, the size of the area searched, or the desired accuracy of the source positions. Then the operator draws the search polygon one vertex at a time until the polygon is closed. The operator may use the visual one meter grid or a loaded map as a guide for drawing the map.

The algorithm then begins decomposing the space into different points with the spacings previously entered. The first, second, and final edges formed by the polygon are then used to create unit vectors along the bottom, right, and left edges, respectively. Then, using the vectors, a reduced search space is created which is smaller by a preset padding value. This padding value guaranties that the robot can reach the outer most points generated by the algorithm should the operator choose to use a wall as a reference when drawing the bounding polygon. The vectors are then scaled to match the spacing values specified earlier. The bottom vector is scaled by the goal spacing factor and the two side vectors are scaled by the row spacing factor. Figure 2a shows the initial state of the exploration algorithm. Here the goal spacing is larger than the row spacing for demonstration. Also note that this exploration polygon is rectangular, however four-sided polygons with sides which are not parallel will also work.

Starting in the bottom left of the padded polygon, the algorithm moves a point back and forth, moving up with each pass. This is accomplished by adding or subtracting the bottom vector. If the next point generated would fall outside the padded polygon, the algorithm will replace it with a point on the edge of the polygon. Then, using the appropriate side vector, the point will be moved up the side. The point then moves back



(a)

(b)

Fig. 2. The exploration plan before (a) and after completion (b).

along a parallel track by once again using the bottom vector. When the point escapes the top edge, the algorithm adds one last pass along the top edge using the bottom vector once again.

Figure 2b shows an example of how the algorithm would finish given dimensions which are not multiples of the row or goal spacing. In this figure, a waypoint would be saved at the tip of each vector. It can be seen in the figure how the algorithm may not have an ideal distribution of points near the edges of the search area. Conversely, the algorithm has successfully covered the area completely. The points generated are now ready to be used to move the robot.

### 4.2. Navigation

The exploration node will begin to move the robot through each waypoint once the operator specifies a starting location. Navigation was implemented via the ROS package named [`move_base`]. A custom supervisory node was used to coordinate robot movement and sample acquisition. The control node's process is shown in Algorithm 4.1.

Algorithm 4.1: Control Node Operation

```
begin
 wait while ROS = OK

 catch movement_finished message
   if movement_finished.status = success
     start sampling
     while sampling = true
       hold position
     end
     store measurement
   end
end
```

The control node is only responsible for maintaining position while the radiation measurement is being taken. The exploration algorithm publishes goal positions to "move base" to move the robot. "Move base" sends a signal when movement is completed. This signal is used to trigger a sample measurement to be taken. This design allows the use of different exploration algorithms for use in different situations.

It can be noted that it is possible that "move base" fails to move the robot to the goal position. In this situation "move base" sends a failure signal instead of a success. When this happens the algorithm simply moves on to the next point. This action was chosen because the location of the robot when "move base" fails is not predicable. The spacing values in the exploration algorithm should be adjusted to account for the possibility of missing measurements.

### 4.3. Source Localization Algorithm

The ultimate task of the autonomous radiation mapping robot is to determine the locations of radiation sources as well as their strength. This can be expressed as the task of determining the sources' parameters. The sources' parameters are their $x$ and $y$ positions and their radiation intensity. It is assumed that the number of sources are known and that the sources being localized will be within the area explored. To solve for the sources' parameters, the measurements from the exploration algorithm could be used to solve the inverse problem. However, due to the dimensionality and uncertainty in the measurements, a deterministic method cannot be used. Instead, a Particle Swarm Optimization (PSO) algorithm was chosen. PSO has the ability to complete a global search in an arbitrary number of dimensions.

In order to implement this algorithm a radiation model and a cost function need to be defined. The algorithm will attempt to minimize the cost function in order to find a global best. The radiation model used in the cost function is given as follows: given $i$ sources with an intensity of $I_i$ at a distance $R_i$ meters from the detector, the measured radiation from one source becomes:

$$M = \sum \frac{I_i}{R_i^2},\ R_i = \sqrt{(X - X_i)^2 + (Y - Y_i)} \tag{1}$$

where $M$ is the measured intensity and $X$, $Y$, $X_i$, and $Y_i$ are the measurement locations and source locations, respectively. This is an application of the inverse square law for radiation modelling.

The PSO will attempt to find the independent variables, in this case the source parameters, which minimize the cost function. The cost function should provide a metric to how well the predicted parameters fit the actual measurement data. This is accomplished by comparing the estimated measurement at each measurement location using Equation (1) to the actual measurement made. The cost is defined as the Root Mean Square (RMS) error between the actual and estimate measurement. The cost function is:

$$Cost = \sqrt{\sum_{j=1}^{m} \frac{(M_{j,p} - M_{j,obs})^2}{m}} \tag{2}$$

where there are $m$ observed intensities ($obs$) which are compared to the predicted intensities ($p$). The PSO method is a population based method. The algorithm initially starts with a population of $N$ particles evenly distributed throughout the decision space. Each particle is a possible solution to the cost function. For the application of source localization each particle will be a column vector with $x$, $y$, and intensity values repeated for the number of sources. Each particle moves around the decision space semi-randomly. Each particle has a velocity which describes its motion and its position is incremented by the velocity vector on each iteration. All of the particles work together by sharing information about their progress. The new velocity is the weighted sum of the previous velocity, the spatial distance (in solution space) to the particle's lowest cost solution (called *pbest*) multiplied by a uniform random variable between 0 and 1 and the spatial distance to the population's best solution (called *gbest*) also multiplied by a uniform random variable between 0 and 1. This moves each particle in a direction which is between its current beast and the population's current best dependant on the random variables. The defining functions for PSO are:

$$v_{(n+1)} = Wv_n + c_1u_1(P_{pbest} - P) + c_2u_2(P_{gbest} - P) \tag{3}$$

$$P_{(n+1)} = P_n + v_{(n+1)} \tag{4}$$

where $v$ is the particle's velocity, $P$, $P_{pbest}$, and $P_{gbest}$ are the positions of the particle, the populations best, and the global best, respectively. In Equation (3), $W$, $c_1$, and $c_2$ are tuning constants and $u_1$ and $u_2$ are random variables between 0 and 1. It is possible to see that the farther a particle is away from either *pbest* or *gbest* the stronger it is pulled toward them, respectively. This causes the particles to pull toward the *gbest* if they are initially far away in the search area from *gbest*. The $W$ is called the inertial value, it encourages the particle to continue on in its current direction. This adds some randomness to the algorithm and makes sure that it will explore a larger area around the personal and global best.

To run the algorithm, a random population of a specified size was generated and each particle would go through a set number of updates with cost function checks at each stage. As discussed previously, each particle is a candidate solution to the source localization problem. In order to initialize the population, each parameter for each source needs to be randomly generated. For each source the $x$ and $y$ position and its intensity must be initialized. The position values are uniformly distributed in the area spanned by the measurement observations inflated by 15%. The intensities are uniformly distributed between 0 and $10^7$ in order to ensure the search space will cover all expected values.

This standard configuration is not robust to different sizes of data or different starting conditions. In order to improve the performance and increase the likelihood that a global minimum is found, a few changes were made to the standard PSO, algorithm.

In order to improve this algorithm, the way that the population communicates was adjusted. In the standard PSO all particles are aware of every other particle. This causes the population to converge quickly, possibly skipping over a better solution. The first change implemented was to limit each particle to only be able to communicate with its neighbour left and right and up and down. This is called a "Grid Connected" swarm. The particles will still converge to a single point but will take longer to do so. This is ideal for discontinuous functions because a minimum may be isolated and difficult to find. It also allowed the use of fewer particles, improving the performance of the algorithm.

The second change was to add a stopping function. This allowed the algorithm to be set with more iterations than necessary without wasting time circling the global best or improving the global best beyond a necessary amount.

The last change was to improve the likelihood that a global best would be found by reseeding the population after it converged. This was accomplished by rerunning the algorithm with the previous global best copied to the new population. The rest of the population would be generated randomly as usual. The stopping condition for when to stop reseeding is when the same global best is found 10 times in a row. This guarantees that the algorithm converges and the possibility of finding the true global best is greatly improved.

After extensive testing, the parameters were determined: $c$ values of 1.49, a $W$ value of 0.72 and a population size of 250 were used. The single run maximum number of iterations was $3,000$. This stops time being wasted on a bad initial population. The results of PSO are displayed in the GUI when the algorithm finishes.

## 5. EXPERIMENTAL RESULTS

### 5.1. Test Plan

In order to test the effectiveness of the completed autonomous radiation mapping robot in an outdoor environment, a test area had to be chosen. An area free from obstacles to localize from would provide a suitable test for the robot localization system. A baseball diamond was chosen for this purpose (see Figure 3). The area was suitably flat for the size of the ARMR and also provided a challenge for the navigation system due to the ease of slipping on the soil surface. In order to test the effectiveness of the

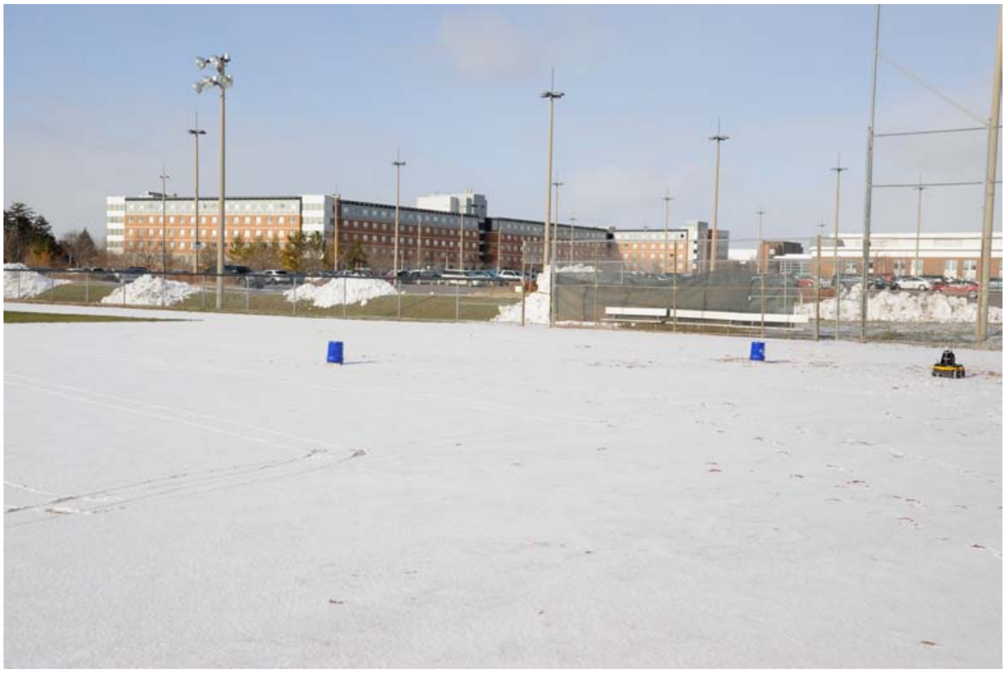
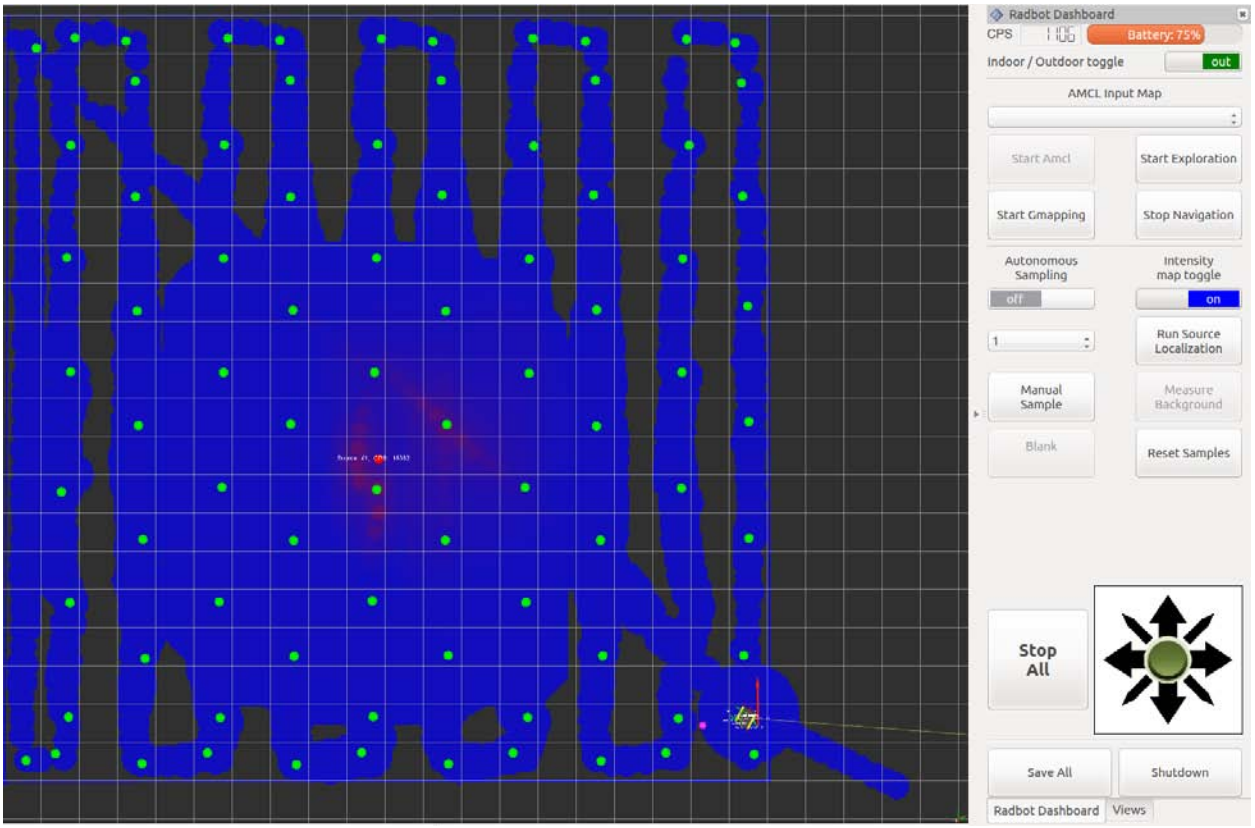


Fig. 3. A view of the two source positions inside the baseball diamond.

source localization algorithm two source configurations were chosen.

First a base station was set up over home plate and surveyed for a period of 45 minutes. A 21 *m* square area was measured within the infield diamond south-west of the base location. Pylons were placed at the corners to be used as a reference for drawing the search area. These proved difficult to view in the GUI and the 1 meter grid displayed in the GUI was used instead to draw a 20 *m* square search area. Figure 4 shows an example of a completed run; the outer blue border represents the explore boundary. Two source positions were selected using the home plate-first base direction as west and home plate-third base direction as south. Using these directions the source positions were measured and marked at 10 meters south by 10 meters west of the north east corner. From now on this position will be referred to as Position 1. The second position (Position 2) was marked at 3 meters south by 10 meters west of the north west corner of the area. A view of the two source locations can be seen in Figure 3. The source positions were then surveyed in the same manner as the base location with a logging time of 20 minutes. The surveyed positions were provided in both latitude-longitude format as well as UTM (Universal Transverse Mercator) format. UTM is a 2D Cartesian earth fixed grid with a spacing of one meter. This makes comparing the results of the source localization system to the surveyed locations possible.

Two tests runs were conducted in the outdoor test environment. Two different source configurations were tested. The first configuration had two of the 1.00 mCi Cs-137 sources placed at Position 2 and one 1.00 mCi Cs-137 source at Position 1. The second configuration had all three of the sources placed at Position 1. The first configuration tested the localization algorithm's ability to localize multiple sources with different radioactivity. The second configuration tests the localization system with one source. For each source configuration the same row spacing was tested. The goal spacing and row spacing were set to 3 *m*.



Radbot Dashboard

CPS 1106

Battery: 75%

Indoor / Outdoor toggle: out

Autonomous Sampling: off

Intensity map toggle: on

Fig. 4. The GUI used in the base station computer.

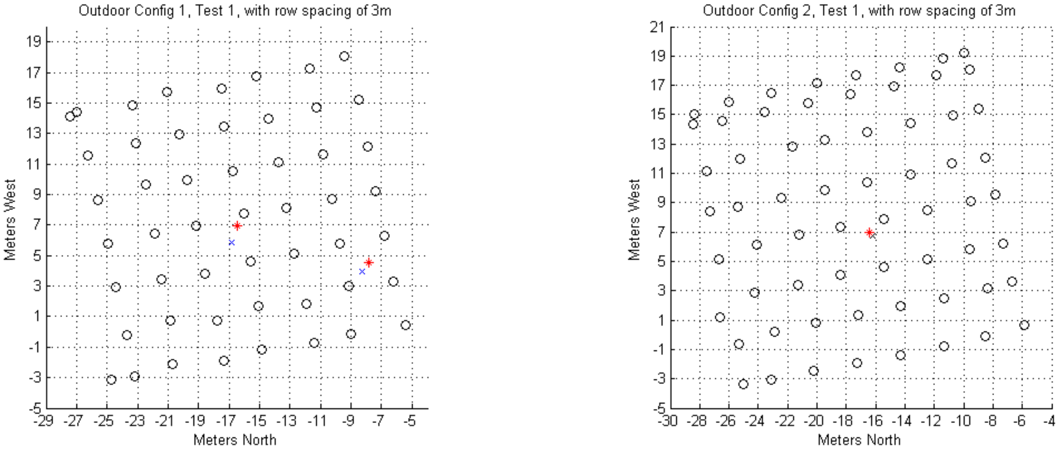
### 5.2. Autonomous Sampling

To start a trial, the ARMR was commanded to search a 20 *m* square area by drawing the bounding box in the GUI on the base station computer and selecting the start location. Once the start location was selected, the robot immediately started moving through the different positions planned by the exploration algorithm. From this point to the end of sampling the ARMR operated autonomously. At the end of each test the robot would return to its start location and then the PSO algorithm was run by selecting the appropriate button in the GUI.

In Figure 5 the sample positions decided by the exploration algorithm for each test are shown as circles. In this figure the axes show the distance in meters from the base station which lay in the north east corner of the diamond. So the points lay south and west of the base station. The coordinates are placed in this fashion due to the ROS standard of having the x-positive direction face north and the y-positive direction face west.

As is visible in Figure 5, the positions are not rigidly aligned to a grid. Conversely the actual positions were recorded instead of commanded positions to account for tolerance in the navigation system and settling time of the GPS sensor.

While the area that the ARMR searched was a fixed size, the number of points that were reached changed based on how the bounding box was drawn as described in Section 4.1. This resulted in each test having a different number of samples. The time it took for the robot to finish was worth noting as well. Table 1 shows the number of samples each test recorded compared to the time taken taking samples. The results show a fairly consistent ~28 seconds per sample. Considering that each position had a sample averaging time of 20 seconds it then follows that the travel time was under 10 seconds per sample.



(a) Configuration 1: 3 meter spacing

(b) Configuration 2: 3 meter spacing

Fig. 5. Estimated source locations produced by the PSO algorithm (x's) with the GPS reference inserted (red stars).

Table 1. Time taken to search a 20 *m* square area.

| Number of Samples Taken vs Time Taken | | |
|---|---|---|
| Test | Number of samples | Time (Excluding Travel Home) (minutes) |
| Config 1 | 51 | 23.17 |
| Config 2 | 64 | 30.07 |

### 5.3. PSO Source Localization Results

Once the ARMR has completed its sampling, the PSO algorithm was run. First the number of sources was specified using the drop down in the GUI (see Figure 4). Then the PSO was run using the appropriate button in the GUI. This ran the algorithm described in Section 4.3.

During testing the runtime of the algorithm was noted. The algorithm was run on a second generation Intel i7 mobile processor. Increasing the number of sources significantly increased the processing time. The runtime changed from an average of 1.03 minutes to an average of 3.42 minutes. The added complexity clearly takes longer to compute.

The positions measured manually with the GPS were used to verify the results of the tests. In order to relate the UTM positions to the results from the PSO algorithm the UTM measurements had to be zero referenced to the base location which was also used as the fixed frame axis in the localization system and for the source localization algorithm.

Viewing the results on the same scale as the search area, the position accuracy is very good, as in Figure 5. In the figures the red stars represent the surveyed source locations and the x's represent the output of the PSO. After the experiments were complete, the results were analysed further.

Table 2 shows the detailed source positions produced by the PSO algorithm. The error displayed is the RMS error for each position. It follows that the total error would increase with more sources, less desirably it seems that the individual errors of each source also increase in scenarios with multiple sources.

The error in the reference measurement and instantaneous GPS error during the test should also be considered. For instance, the RMS error of the base station position was 0.163 *m* and the error for Position 1 0.226 *m*. These values represent highly accurate measurements which were the result of post processed data from long stationary recordings. It is then reasonable to assume that the live GPS measurements combined

Table 2. RMS error of each test when compared to the GPS reference.

| PSO Results vs GPS Surveyed Points | | | | | | |
|---|---|---|---|---|---|---|
| Test | Position 1 | | RMS error | Position 2 | | RMS error |
| | $x$ $(m)$ | $y$ $(m)$ | $(m)$ | $x$ $(m)$ | $y$ $(m)$ | $(m)$ |
| Reference | -16.4535 | 6.9396 | | -7.8704 | 4.5126 | |
| Config 1 | -16.8423 | 5.8850 | 1.1240 | -8.2908 | 3.9889 | 0.6715 |
| Config 2 | -16.2191 | 6.7112 | 0.3272 | | | |

with a non-deterministic algorithm could result in errors of the range displayed in Table 2. however, from a practical standpoint, the results are accurate enough for use in a real-life situation.

Table 2 only shows part of the results. The algorithm also reports the assumed Counts Per Second (CPS) at one meter from each source. Table 3 shows the output from the PSO algorithm. It is clear that for configuration 1 the radiation intensity at Position 2 is about twice that of Position 1 which is what was expected. Configuration 2 shows an intensity roughly the sum of the CPS of the first tests. These results show that the PSO algorithm is producing reasonable accuracy for each source.

Table 3. Source intensity reported by the source localization algorithm.

| PSO Source Intensity Results | | |
|---|---|---|
| Test | Position 1 (CPS at 1 $m$) | Position 2 (CPS at 1 $m$) |
| Config 1 | 6345 | 10751 |
| Config 2 | 14076 | |

## 6. CONCLUSIONS

A fully autonomous radiation mapping robot was developed and tested. In this implementation the task of producing a navigation plan was accomplished by decomposing an input polygon into a grid based on parameters set by the operator. The radiation source parameter estimation was accomplished by a modified Particle Swarm Optimization (PSO) algorithm. The test results show the effectiveness of this system in a real-world scenario. The results confirm that the full system works as designed and produces an accurate representation of the source configuration. The PSO algorithm was able to produce estimates in under five minutes for a two-source configuration.

## ACKNOWLEDGEMENTS

The authors would like to thank the University Network of Excellence in Nuclear Engineering (UNENE) and the Natural Sciences and Engineering Research Council (NSERC) of Canada for their financial support of this research.

## REFERENCES

1. Hykes, J.M. and Azmy, Y.Y. “Radiation Source Mapping with Bayesian Inverse Methods.” *Nuclear Science and Engineering*, Vol. 179, No. 4, pp. 364–380, 2015.
2. Jarman, K.D., Miller, E.A., Wittman, R.S. and Gesh, C.J. “Bayesian Radiation Source Localization.” *Nuclear Technology*, Vol. 175, No. 1, pp. 326–334, 2011.
3. Chin, J.C., Yau, D.K.Y. and Rao, N.S.V. “Efficient and Robust Localization of Multiple Radiation Sources in Complex Environments.” *Proceedings of the 31st International Conference on Distributed Computing Systems*, pp. 780–789, 6 2011.

4. Cortez, R.A., Papageorgiou, X., Tanner, H.G., Klimenko, A.V., Borozdin, K.N., Lumia, R. and Priedhorsky, W.C. “Smart radiation sensor management.” *IEEE Robotics & Automation Magazine*, Vol. 15, No. 3, pp. 85–93, 2008.
5. McDougall, R.D., Nokleby, S.B. and Waller, E. “Experimental Testing of a Probabilistic-Based Radiation Mapping Robot.” *Proceedings of the 2013 CCToMM Symposium on Mechanisms, Machines, and Mechatronics*, pp. 1–7, 2013.
6. Mcdougall, R. *Robotic Radiation Mapping Usin Modeling and Probabalistic Analysis of Sparse Data*. Ph.D. thesis, University of Ontario Inst. Tech., 2015.
7. Medina, A.J.R., Pulido, G.T. and Ramírez-Torres, J.G. “A Comparative Study of Neighborhood Topologies for Particle Swarm Optimizers.” In “IJCCI 2009 - Proceedings of the International Joint Conference on Computational Intelligence, Funchal, Madeira, Portugal, October 5-7, 2009,” pp. 152–159, 1 2009.
8. Langley, R.B. “RTK GPS.” In “GPS World,” , 9 1998.
9. Takasu, T. and Yasuda, A. “Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB.” In “Proceedings of the International Symposium on GPS/GNSS,” , 11 2009.