GIS Map Based Mobile Robot Navigation in Urban Environments

J.M Mirats-Tur, C. Zinggerling and A. Corominas-Murtra

Abstract— **Mobile robots need information about its spatial environment to solve any task in a robust, reliable and continuous form. This is usually encoded in a so-called *map*. We propose the use of a standard, human-readable existing GIS map for robot navigation in urban scenarios. A centralized GIS map server architecture is presented to which a robot or team of robots can connect by using a developed communication protocol. The system is scalable to large environments and large groups of robots. Furthermore it demonstrates the possibility for a robot to accurately navigate by using an already existing map which has not been constructed using the robot own sensors. Details of the implemented system architecture as well as a position tracking experiment in a real outdoor environment, a University Campus, are provided.**

I. INTRODUCTION

For a mobile robot to succeed with the assigned task it is necessary to have information about its spatial environment, often encoded in a so-called *map*. Whichever the application, the robot needs the ability to reach any goal from any start point, i.e., to navigate. In other words, the robot must have some kind of knowledge about the relative position of other objects in the space they share, in order to, for instance, be able to interact with these objects (think in manipulation or transportation) or to calculate a path to a desired position.

To this aim, there are mainly two approaches. The first is based on the assumption that the robot has no idea about its environment, solving a problem of *map learning*, i.e., obtaining a suitable representation for the data acquired by the sensors during an exploration phase. This is closely related to localization leading to a question of the kind: who was the first, the egg or the chicken? A map is needed to perform localization, but, on the other hand, the robot position is needed to build the map. This challenge is known as *SLAM* (Simultaneous Localization and Mapping) existing a vast literature about it [2]. The second approach is based on the assumption that the robot will perform its tasks in an already somewhat mapped environment. So the robot is provided with an initial, surely partial and incomplete, knowledge about the environment contained in a map which has not been constructed using its own sensors. This kind of map based navigation is natural for humans: when visiting an unknown city, we just need an existing map in standard format to navigate as defined above.

This work has partly been supported by Cicyt Navrob project DPI2004-05414 of the Spanish Science and Education Ministry and European STREP project URUS with reference IST-045062.

J.M Mirats-Tur and A. Corominas are with Institut de Robòtica i Informàtica Industrial, IRI (UPC-CSIC). Barcelona, Spain. jmirats,acorominas@iri.upc.edu

C. Zinggerling is with the Centro Internacional de Métodos Numéricos en la Ingeniería, UPC, Barcelona, Spain. czingger@cimne.upc.edu

Two map representations are commonly used, topological and metric, although the trend is to combine both leading to a vast class of hybrid maps, see [6] for a review. In topological maps the environment is stored with linked nodes. They contain distinctive places [9] and the connexions between them, without metric information in its basic form [17]. Metric maps store unambiguous locations of objects, usually in a 2D frame, which allow to precisely position them. Objects may be stored from different points of view: they can be considered as punctual [5], as different points recorded from a surface [16], corners with an associated orientation [7] or lines defining polygonal boundaries [8]. Alternatively, they can contain a free space representation, i.e., the portion of the environment that is accessible to the robot. This is the main idea behind occupancy grids [12], [13].

Readable human maps offer advantages, specially when human-robot interactions are expected. Although the use of a map by a person requires high-level cognitive functions, partially solved for robots, a map model organized within a hierarchy and accepting semantic information will be more suitable to interface with humans. Spatial representations such as Geographical Information Systems (GIS) based maps can use already developed interfaces as a powerful human-robot interaction tool, specially in urban areas [10]. We describe in this paper the use of such standard human-readable maps for mobile robot navigation in urban zones. The issues of map-sharing and communications so that a robot or a team of robots can connect to this map and use it to navigation purposes are also tackled.

This work is organised as follows. Section II explains the used robot map format. The GIS map translation to a robot-readable format is given in section III. The designed architecture is explained in section IV and some experimental results from using the whole proposed architecture are shown in section V. Some conclusions are outlined in section VI and, finally, an appendix is given with a brief review on GIS.

II. ROBOT MAP MODEL

Our application, within the URUS project [14], is envisaged for multirobot teams providing services in urban environments such as a university campus or a city quarter. Hence, we need to use human compatible maps, as those provided by GIS, for robot navigation. Since the system is to be used in large scenarios we are want to assure scalability up to large environments keeping in reasonable bounds both, memory and computational resources. Compactness also gains importance when whole or parts of the map have to be sent through a communication network. Hence, we want the robot map to fulfill the following requeriments: scalability, accuracy, flexibility, three dimensional description, automatic conversion from a GIS source and human compatible, [3].

The used representation is based on geometric entities and inspired from the 'GIS vector' format [10]. Height information of geometric entities is added to provide simple 3D information. Stairs and ramps are modelled as well, since they are key 3D obstacles for navigation in outdoor areas.

The map $\mathcal{M}$ is defined by the coordinates limiting its borders together with a list of NB obstacles. The left-up corner is (mx_1, my_1) while the right-down (mx_4, my_4).

$$\mathcal{M} = \{mx_1, my_1, mx_4, my_4, o^1, ..., o^{NB}\}$$

The $k-th$ obstacle in the map, o^k, is defined with a list of NS^k segments, an integer id^k identifying the obstacle, an integer ST^k describing the type of shape representing the obstacle and related semantic information, $semanticI^k$.

$$o^k = \{s^k_1, ..., s^k_{NS^k}, id^k, ST^k, semanticI^k\} \quad k = 1..NB$$

ST indicates whether an obstacle is represented with a closed or open polygon, with a closed or open curved shape, or it represents stairs or a ramp. The semantic information is a character string labelling some features of the obstacle as if it is a building, a column, a trash, etc.

The $l-th$ segment of the $k-th$ obstacle , s^k_l, is defined from points a^k_l to b^k_l. Height, h^k_l, an indoor/outdoor boolean value, $inOut^k_l$, and semantic information (as if it represents a wall, a door, the material from which is built, etc.) complete the segment definition. All segments are oriented, so they are defined from left to right viewed from the free space.

$$s^k_l = \{ax^k_l, ay^k_l, bx^k_l, by^k_l, h^k_l, inOut^k_l, semanticI^k_l\}$$
$$k = 1..NB, \; l = 1..NS^k$$

Stairs and ramps are also modelled in our map description, introducing 3D information. *Stairs* are modelled as a list of segments. From downstairs, the first segment is oriented from left to right, with a height equal to that of the first step, just as a 'short' wall. Next segments are equally oriented but far away the step width and with the corresponding height. Finally, a segment oriented from left to right with zero height ends the stair. Fig. 1 shows the stair and ramp models.

Ramps are modelled as obstacles with null height. Ramp borders are described by the closed polygon formed when projecting the ramp onto the 2D plane. Ramp orientation is parameterized with the normal vector to the surface.

We notice how the designed representation scales up to large environments thanks to the implicit 'metric/topologic grouping' with geometric entities. In the run experiment (see section V), an area of about $10000m^2$ has been described with a file of $30KB$. Without any data compression or encoding, the compactness figure, which can be measured as a ratio of $bits/m^2$, is about $3Bytes/m^2$, much lower than using a directed graph (requiring around $30Bytes/m^2$) or a grid map [1].

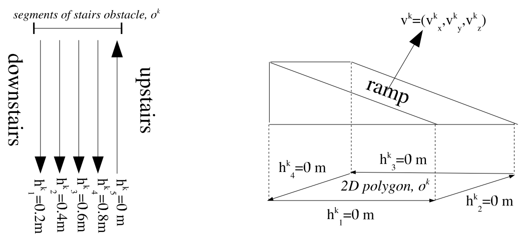

Fig. 1. Stairs and Ramp model.

III. From GIS to robot map

A computer acting as GIS server contains a map of the whole urban area where the robot, robots, or different teams of robots are supposed to develop their tasks. Hence, given a GIS map from a determined urban zone, the first stage is to translate it to a robot understandable format.

One of the server main functions is to manage, control and attend each of the robots that have connected to it. The server has an agent which, in real time, directs each robot towards accomplishing the assigned tasks. Examples of server functions are to provide maps of the nearby zone of each robot, to program a route to follow, to specify a destination point, to make an on screen localization of the robots position, to control possible anomalies or to annex new elements to the cartography from robot sensor readings. The GIS database is computationally expensive, hence the centralized architecture. Furthermore, we can share important algorithms such as performing map updating from robot sensor readings or global path-planning for multiple robots.

A. Map Generation

So the GIS server provides the robot a map of an area centered on its position. This is obtained converting a standard GIS (vectorial) format to a text file where the geometry is translated to a robot-readable format as shown in figure 2.

```
border;
-4;100;
100;-4;
#
obstacle;
1;1;4;column,A5;
#
4.53;12.10;
7.29;12.10;
5;1;wall,orange,brick;%l1
#
obstacle;
110;2;1;wall,A4;
#
-3;0;
-4;0;
1.2;1;wall,grey,cement;
#
```

Fig. 2. Example of the generated text file containing the map in a robot understandable fashion.

Since GIS is a multilayer system, each of the layers loaded in the general map must be checked to determine which elements are of interest to generate the partial map that will be sent to the robot (only layers of area and lines will be used). Thanks to one of the main characteristics of GIS, each object being associated to alphanumeric data, the information related to each element is used to group the geometry. The robot map is provided at the request of a given robot under two determined circumstances:

- When the robot has to auto locate itself for the first time and sends to the server a *Ready* command (Refer to table II for a description of the currently implemented robot requests). In this case it receives the whole map.
- When the robot directly requests a map using the *GetMap* command.

Once the map is obtained by coding the GIS map, it is sent to the robot through the network using the command *SetMap(M)*, where M is the file coded in binary format.

B. Construction process of the partial map

The map sent to the robot is automatically generated in the GIS server. The procedure is as follows:

- Delimitate the zone to be sent: H x W area around the current position of the robot.
- Translate the geometry of each one of the objects within each layer to the text format.
- Add the alphanumeric information related to each object to be included in the map.
- Group the geometry by type of obstacle.
- Build the map in text format and encapsulate it in binary format to be sent to the robot through the network.

IV. System definition and implementation

A server-client scheme has been designed, where the computer acting as GIS server provides map services to any robot client requiring it. Communication between robots and the GIS server is done via point to point wireless ethernet connections. Communications are supposed to exist during the whole operation of the robots, it is beyond the scope of this paper the study of robot formations so as to maintain stable communications with the central GIS server.

Once the connection is set up between a robot and the GIS server, we have to provide means for information exchanging between them. This has been accomplished by means of a Data Base Management System (DBMS). Our implementation uses SQL (Structured Query Language), which is a computer language aimed to store, manipulate, and retrieve data stored in relational databases [11], so it is perfectly suited to implement a bridge between the GIS software containing the map and the robot navigation algorithms. Each of the robots is provided with user and password identification in order to stablish the connection to the database. This fact, in addition to a security issue, gives the possibility to more than one robot to access the same GIS map.

From the robot point of view, DBMS managament is provided by a public C++ library called MySQL++ which may run under Linux or Windows operating systems and is easily integrable with the onboard algorithms. Main required parameters for a robot to stablish a connection with a GIS server are: the name of the database, we may have more than one database in the same computer server, the host name or IP address of the DBMS server, the server address in private wireless network, the user name and password to log in.

To maintain the system scalable, robust and flexible, a *plug and play* procedure to add new robots has been designed. Suppose a new robot enters an existing multi-robot team to aid for some tasks. The robot should notify its existence to the server to be able to use the GIS map, and hence sharing the spatial knowledge together with the rest of the robots. When the robot has checked for all its sensors and has started all the required internal algorithms, it sends a $Ready$ signal to the server together with a random generated number. Then, it waits for server acknowledge and an assigned identity tag. The robot will then use this assigned id to communicate with the server until it finishes its tasks. Also, in order to mantain an active status of a robot into the server system, each robot sends periodically an *alive* message.

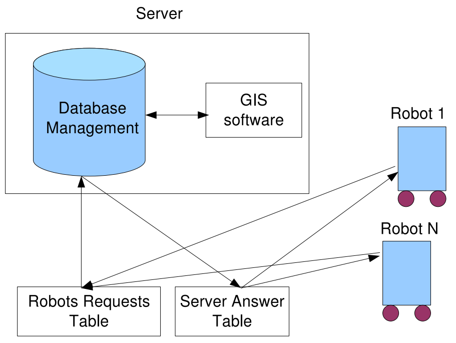

Fig. 3. Simplified description for the designed communication architecture between robots and GIS server.

At this point, a logical connection has been stablished among a robot and the GIS server, so information exchange between them may proceed. This is accomplished by using two tables, one for robots requests, the other for server answers. Different procedures are designed for operation with both tables. On the robots requests table (RR) only robots are allowed to write, while the server may perform read and delete operations. Each RR table row will contain a unique robot request. The procedure is as follows, a robot requests a specific operation, see table II, so a new entry in the RR server table is generated. The server then reads the new entry and whether processes or stores it in secondary tables for internal computations which are not accessible from the robots (think for instance in having implemented a global path planning algorithm in the GIS server). Then, the server delete this entry from the RR table and may proceed to the next robot request.

TABLE I

PERMITTED OPERATIONS FOR ROBOTS AND SERVER

	Robots Requests Table	Server Answers table
Read	Robot	Server
Write	Server	Robot
Delete	Server	Robot

On the other hand, the server answer table, SA, can only be written by the server while robots can only perform read and delete operation on it. Every row will contain a unique answer which will be deleted by the corresponding robot once read. Table I summarises the permitted actions for robots and server while tables II and III summarise the currently implemented functions.

TABLE II
CURRENTLY IMPLEMENTED FUNCTIONS FOR ROBOTS REQUESTS

Operation code	**Parameters**	**Description**
Ready	Magic number	Indicates the robot is ready to start tasks and listen to the SA table
Sttoped		Robot is stopped
Alive		Robot is alive, periodically sent
OpAck		Robot acknowledges for the received operation from the server
SendPos	x,y,θ and robot vel	Robot sends position estimation to server. To be sent periodically
SendGPS	Lat, Long	Robot sends GPS Latitude and Longitude information
SendImage	I - binary image, x,y and ID	Robot sends a binary encoded image (I) grabbed at position (x,y) associated to an object map (ID)
GetMap	x,y, A	Robot asks for new map with area A centered on its current (x,y) position
GetPath		Robot asks for a new path to complete the actual task. The server knows actual position and destination
GetPosIni		Robot asks for initial position
GetPosEnd		Robot asks for an end position of a track
SetObject	x,y, ID	Robot has observed an object identified as ID in position (x,y) that has to be included within the map

Just as an example of how these commands work, consider the case of controlling and tracking the robot position. The robot indicates to the GIS server its position at every moment through the communication protocol by means of the command $SendPos$. When the agent on the server detects an entrance of this command, draws the position in the GIS map. The system draws and stores the trajectory of the robot from the origin to the actual location to verify and control the tasks made by each robot at every moment. In the case one robot leaves its original trajectory or if it is desired to reassign a new one, we can abort the current assigned task by means of the command $Abort$. The command $SetPosEnd(X, Y)$ is used to indicate a new destination.

V. EXPERIMENTS

A real experiment demonstrates the functioning of the designed approach. The used map describes the surroundings of the Computer Science School at the Campus Nord of the UPC, an outdoor scenario of about $10000m^2$ being the test bench scenario for the URUS project [14]. Fig. 4 shows the geometric part of the map, the considered origin and the coordinates of two illustrative points. Pictures shown in Fig. 5 were taken from locations marked with blue arrows in Fig. 4. This gives an idea of the challenging scenario we are facing to, where robots are supposed to navigate in a robust and continuous manner.

TABLE III
CURRENTLY IMPLEMENTED FUNCTIONS FOR SERVER ANSWERS

Operation code	**Parameters**	**Description**
Start		Server is ready to listen to the RR table
OpAck		Server acknowledges the received request from the robot
Abort		Server indicates the robot must abort its current task. User supplied for supervisory purposes
SetMap	M	Server has sent the robot the requested map. M is the binary coded file sent
SetPath	(x_i, y_i)	Server sends the robot a new path to complete the current task
SetPosIni	(x_{ini}, y_{ini})	Server gives the robot an initial position of a track to be run
SetPosEnd	(x_{end}, y_{end})	Server gives the robot an end position of a track to be run

Our interest with this experiment is twofold. First, to evaluate whether or not the robot is able to navigate in urban environments using a human-like map which has not been constructed using its own sensors. Second, to test the system implementation as defined in the previous sections. The experiment consisted in performing a close loop around building A5 (see Fig. 4), accounting for a path length of approximately 150 meters. The employed tracking method was a particle filter. The odometry of the robot was used to propagate the particles while data from a 2D laser scanner was used to compare information stored in the map with robot observations; this comparison was used to update the probability of the considered position hypothesis for each particle. A detailed development of the used navigation approach is provided in [4].

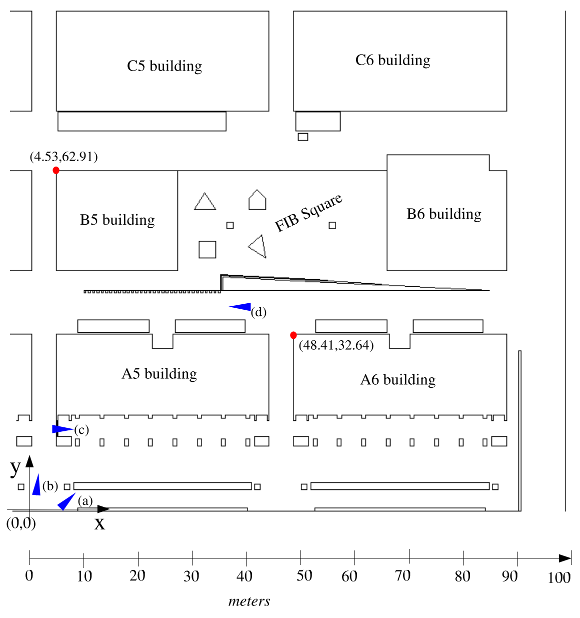

Fig. 4. Map geometric part of the surroundings of the 'FIB Square' at the Campus Nord of the Universitat Politècnica de Catalunya (UPC)

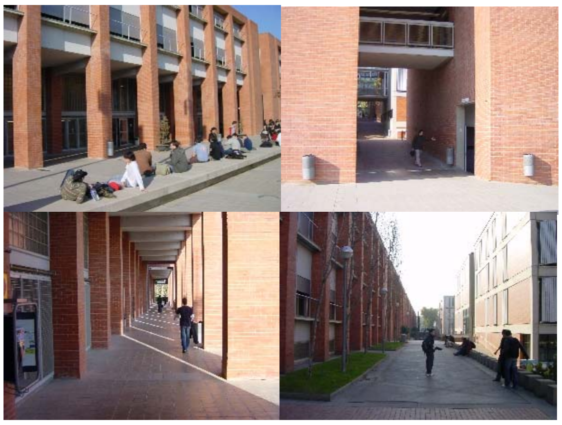

Fig. 5. Pictures taken from a,b,c,d points on Fig. 4 representing the path run by the robot in the performed experiment.

The set up for this experiment included a central computer with the GIS server, a single robot and a point to point wireless connection between them. When the robot is turned on, it connects to the server following the procedure given in previous sections and it requests a portion of the general GIS map (we are assuming the robot is already localized in the environment). When the user on the GIS server sets a goal, the robot starts running to it. Apart from the local navigation algorithm on the robot, comunication to the GIS server occurs in order to update the robot position in the server and to provide the robot with a proper map to continue its path, depending on its position. Results of this experiment are shown in Figs. 6 and 7 where some snapshots representing the robot position as respectively seen by the GIS server and the robot itself are shown.

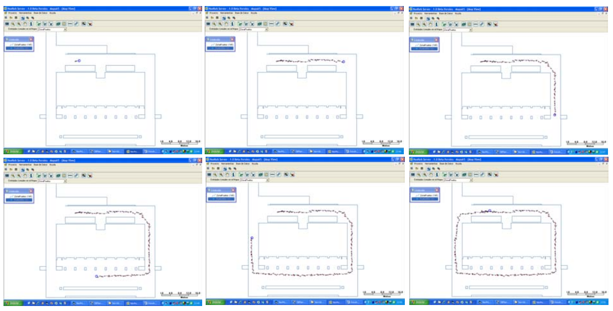

Fig. 6. Some captions of the reported robot position within the GIS server. The robot sent its position to the server at 3Hz, and operations on tables RR and SA were performed as described.

VI. Conclusions

More and more Society is demanding new robotic applications in which robot performance is closer and closer for humans to understand. If we think in the use of mobile robots in common human environments, as urban settings are, for common tasks, as may be for instance surveillance, merchandise delivering or garbage collection, we will see the need for a robot or team of robots solving a given task to use or re-use an existing map of a big urban-like

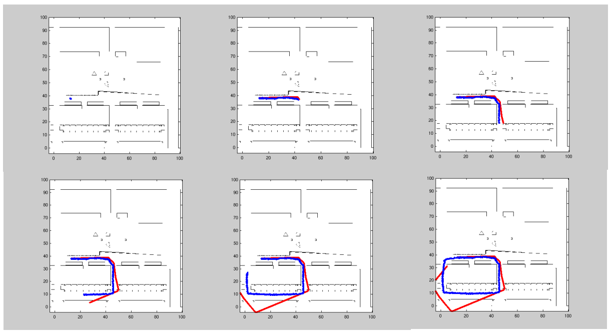

Fig. 7. Some captions of the robot position seen locally by the robot. Blue line is for the output of the tracking filter while red line is pure odometry

area. Yet, better if this existing map can be in a human-understandable fashion, since we will surely need to have human-robot interaction in order to specify tasks and places to go, for instance in a big city quarter.

This is the issue we have mainly tackled in the present paper. A server-client architecture has been designed in which the server contains a GIS based map of the area, commonly large, where a team of robots is supposed to work, and the clients, actually a robot or team of robots, can connect to the general data base in order to retreive a part or the whole of the map needed to perform their tasks. Clients can, at the same time, provide up to date information in order to maintain updated the general data base. So, given a large area to be covered by a team of robots for which a human-like map format exists, we have provided a method that allow this team of robots to safely and robustly navigate in this large area. A communication protocol has been presented between the GIS server and the robot clients tht used DBMS and SQL in its implementation. A real experiment in a common outdoor environment, a University Campus, has been performed with a single robot in order to validate the GIS based navigation system using point to point wireless network technology.

Appendix

A Geographical Information System (GIS) can be defined as a geographic information handling technology. It allows to manage a series of spatial, geographic, data and to make complex analyses [18]. In recent years the importance of GIS has considerably grown since getting reliable geographic data is becoming more and more essential due to the large economical and time investment needed to obtain these data.

GIS deals with geographic plus semantic data, and hence it is being used in most of the information management systems which depend on the location of objects in given surroundings [10]. Particularly, a GIS may be helpful when it is desired to give autonomy and human-like resolution capabilities to a robot or team of robots in non-controlled surroundings, i.e, the robot knows the terrain in which it moves and hence could take decisions in advance [15]. Also, a GIS system can help to maintain control of the robots accurately indicating on a map their position.

A. GIS main characteristics

GIS evolves from the combination of classical paper maps and modern CAD systems. They have been developed to offer digital cartography integrated to alphanumeric information. They are made up of layers each one displaying information related to a specific group of data (for example, parcels, blocks or streets).

There are different forms to model the connection between geographic and topological objects. Depending on how this connection is carried out two types of GIS are defined: Vectorial and Raster GIS. The first, uses vectors defined by pairs of coordinates related to some cartographic system to compose each layer of the map. The second, displays and stores drawings or images like a dot matrix (cells).

Applications of GIS are widely open, including management environment, logistics, urbanism and transport or distribution networks (water, energy...) among others [10]. The application range can be divided into three main classes:

- As a tool for map building.
- Support for spatial analysis.
- Geographic data base, with functions of storage and space information retrieval.

According to existing literature, a system is named like a GIS depending on its capacity to perform whether queries or spatial analysis. This means that any data or object within a GIS can geographically be located by its coordinates and/or descriptive attributes. These attributes provide a powerful tool, since they can relate an existing geographic object to an alphanumeric data set: a fundamental characteristic of GIS.

B. GIS Implementations

There are several types of available GIS, commercial as well as open source codes. Esri's ArcGIS [19] and Intergraph GeoMedia [20] are two of the up to date most used commercial applications. Within the Open Source applications we can find MapServer [21] and Grass [22], widely used by scientists. As a matter of curiosity we outline the following:

- ArcGIS is an integrated collection of GIS software products for building a complete GIS [19].
- GeoMedia provides a full suite of powerful analysis tools, including attribute and spatial query, buffer zones, spatial overlays, and thematics [20].
- MapServer is an Open Source development environment for building spatially-enabled internet applications; is not a full-featured GIS system, nor it does aspire to be. Instead, MapServer excels at rendering spatial data (maps, images, and vector data) for the web [21].
- GRASS is a GIS used for geospatial data management and analysis, image processing, graphics/maps production, spatial modeling, and visualization [22].

References

[1] K.O. Arras, R. Philippsen, N. Tomatis, M. de Battista, M. Schilt and R. Siegwart, "A Navigation Framework for Multiple Mobile Robots and its Application at the Expo.02 Exhibition", *in Proc. of the 2003 IEEE International Conference on Robotics and Automation, ICRA*, Taipei, Taiwan, September 2003.

[2] T. Bailey and H. Durrant-Whyte, Simultaneous Localisation and Mapping (SLAM): Part II-State of the Art, *Robotics and Automation Magazine*, vol 13, 2006, p. 108-117.

[3] A. Corominas-Murtra and J.M. Mirats-Tur, *Map Format for Mobile Robot Map-based Autonomous Navigation*, Technical Report IRI-DT 2007/01. February 2007.

[4] A. Corominas-Murtra, J.M. Mirats-Tur and A. Sanfeliu, Active Global Localization for Mobile robots in cooperative and Large Environments, *Robotics and Autonomous Systems*, Volume 56, Issue 10, pages 807-818, October 2008.

[5] H. Feder, J. Leonard and C. Smith, Adaptive mobile robot navigation and mapping, *International Journal of Robotics Research*, 1999, 18(7), pp. 650-668.

[6] D. Filliat and J-A. Meyer, Map-based Navigation in Mobile Robots. I. A review of localization strategies, *Cognitive Systems Research*, Vol 4, Issue 4, December 2003.

[7] A. Grosmann and R. Poli, Robust mobile robot localisation from sparse and noisy proximity readings using Hough transform and probability grids, *Robotics and Autonomous Systems*, 2001, Vol. 37, No. 1, pp. 1-18.

[8] L. Iocchi and D. Nardi, Hough Localization for mobile robots in polygonal environments, *Robotics and Autonomous Systems*, 2002, Vol. 40, No. 1, pp. 43-58.

[9] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon and F. Savelli, "Local Metrical and Global Topological Maps in the Hybrid Spatial Semantic Hierarchy", *in Proc. of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA, USA, 2004.

[10] J. Maantay and J. Ziegler, *GIS for the urban environment*, Environmental Systems Research Inst., 2005.

[11] J. Melton, *Understanding the New SQL: A Complete Guide*, Morgan Kaufmann publishers, 1993.

[12] H. Moravec and A. Elfes, "High resolution maps from wide angular sensors", *in Proceedings of the IEEE international conference on robotics and automation*, IEEE Press, pp. 116-121, 1985

[13] M. Ribo and A. Pinz, A comparison of three uncertainty calculi for building sonar-based occupancy grids, *Robotics and Autonomous Systems*, 2001, Vol. 35, No. 3-4, pp. 201-209.

[14] A. Sanfeliu and J. Andrade-Cetto, "Ubiquitous networking robotics in urban settings", *in Proceedings of the IEEE/RSJ IROS Workshop on Network Robot Systems*, pages 14-18, Beijing, October 2006.

[15] C. Tao, C. Vicent and J. Li, *Advances in mobile mapping technology*, Taylor and Francis, 2007.

[16] S. Thrun, W. Burgard, D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping", *in Proceedings of the IEEE international conference on robotics and automation*, 2000, pp.321-328.

[17] C.J. Tylor and D.J. Kriegman, Vision based motion planning and exploration algorithms for mobile robots, *IEEE Transactions on robotics and Automation*, 1998, 14(3), pp. 417-427.

[18] F. Wang, *Quantitative methods and applications in GIS*, CRC Pr i Llc. 2006.

[19] http://www.esri.com

[20] http://www.intergraph.com

[21] http://mapserver.gis.umn.edu/

[22] http://grass.itc.it/