Proceedings of the 8th IFAC International Conference on
Manoeuvring and Control of Marine Craft
September 16-18, 2009, Guarujá (SP), Brazil

Occupancy Grid Mapping in an Underwater Structured Environment ⋆

E. Hernández * P. Ridao * A. Mallios * M. Carreras *

* *Department of Computer Engineering, University of Girona, 17071 Girona, Spain (e-mail: {emilihb,pere,amallios,marcc}@eia.udg.edu).*

Abstract: This paper presents practical results about the occupancy grid mapping of an underwater man-made environment using a sensor suite commonly available in nowadays Autonomous Underwater Vehicles (AUVs). The proposed algorithms are tested to be incorporated as part of the design of a new motion control system to integrate reactive obstacle avoidance with local path planning techniques to provide safe real-time guidance capabilities. The paper focus on the use of a sonar scan matching improved dead-reckoning navigation (Doppler Velocity Log (DVL) and Motion Reference Unit (MRU) based) together with an standard occupancy grid mapping algorithm. A conventional inverse sensor model for a sonar profiler is used and compared against a new inverse sensor model proposed to take advantage of the use of widely available imaging sonars. The system is validated experimentally on a dataset gathered with an AUV guided along a 600m path within a marina environment.

Keywords: Robot localization, scan matching, map building.

1. INTRODUCTION

When an AUV carries out a mission, the robot has to be able to find a safe path from a starting point to an end point while avoiding the obstacles. To do so, it is required to use a motion system which is a combination of software elements that work together to generate collision-free trajectories to move a robot through the scenario in which a mission takes place. Usually, an AUV requires two motion systems. One computes a global trajectory for the whole execution. The second one decides how to move the robot between two points on the global path. Fig. 1 illustrates a possible motion system design based on Mínguez et al. (2004).

The global motion system is usually organized in two modules: the model and the planner. The model module is in charge of modeling the environment using the previous knowledge of the scenario (i.e. a bathymetric map in underwater robotics) or based on sensor data. The planning module contains a global path planner that uses the model representation in order to find the trajectory that best suits the accomplishment of the mission objectives.

The local motion system is the part that generates collision-free trajectories between successive positions on a given global path. The design of this system is determined by several factors like the scenario model, the deliberative planning and the motion generation. It has three modules: a local model builder, an obstacle avoidance system and a local path planner. The interaction between the modules may differ depending on the information available and the tools needed to implement each module.

⋆ This research was sponsored by the Spanish government (DPI2008-06545-C03-03) and FREEsubNET (MRTN-CT-2006-036186).

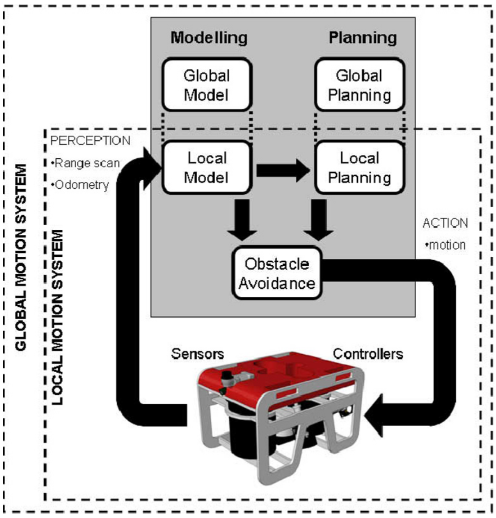

Fig. 1. Full motion system.

The local model builder maps the environment based on sensorial information of the vehicle and provides two outputs: 1) the robot's position within the local map and, 2) the local map itself. The obstacle avoidance module computes a reactive collision-free motion based on the map content and the output of the planner. The local path planner objective is to find a path between the actual robot pose and the local goal (projection of the global goal onto the local map) through the free space. The use of deliberative path planing at this level is useful in avoiding cyclical motions and trap situations.

This paper is focused on the first module, proposing an uncoupled solution for the navigation and mapping problems within local map builder. Although during the last years great efforts have been dedicated to provide a coupled solution to the so-called Simultaneous Localization And Mapping (SLAM) problem, still it is a hard problem to solve, in particular for unstructured underwater scenarios with poor visibility. Hence, taking into account that the motion systems is aimed at achieving a safe guidance of the AUV in the presence of obstacles, which means it has to run in real-time, and assuming that only a small portion of the area surrounding the AUV has to be mapped, it is possible to uncouple both problems. Hence, following the studies carried out in the mobile robotics domain, in this paper we use the MSISpIC, a probabilistic sonar scan matching algorithm (see Hernández et al. (2008)) to improve the DVL-MRU based dead-reckoning navigation. MSISpIC is particularly suitable to be used with mechanically scanned profiling or imaging sonars commonly available in nowadays ROVs and AUVs. Map building is addressed through the use of an standard occupancy grid mapping algorithm. For the mapping two inverse sensor model are used, a conventional one broadly used for range finders in mobile robotics and a proposed new one adapted to the particularities of imaging sonar sensors.

The paper is structured as follows. In section 2 it is described the MSISpIC algorithm. Map building process using the occupancy grid mapping algorithm is explained in section 3. Section 4 reports the experimental results and section 5 exposes the conclusions.

2. SCAN MATCHING

Scan matching techniques estimate the relative robot displacement between two configurations by registering the overlap between two consecutive range scans normally gathered with a laser or a sonar sensor. Usually, these techniques are based on a two steps iterative process which is repeated till convergence. The robot displacement is computed by approximating the solution to the best overlap between two scans looking for the closest point for each single data of the scan. After that, a minimization process to estimate the solution is done. This process is repeated until convergence.

In this work, the navigation is solved using the MSISpIC algorithm presented in Hernández et al. (2008) which is an extension to the pIC method (Montesano et al. (2005)) but adapted to be used with mechanically scanned imaging or profiling sonars.

The pIC algorithm is a statistical extension of the ICP algorithm (Besl and McKay (1992)) which is able to deal with sparse sonar data. Nevertheless, pIC was designed to be used with range scans gathered in one shot and hence, it cannot be applied to slow mechanically scanned sonars like those common in underwater applications. MSISpIC is a natural extension of the previous algorithm which compounds the robot trajectory with the range and bearing data of all the beams, represents them and their uncertainty in a unique reference frame and then applies the standard pIC.

More precisely, the MSISpIC (Algorithm 1) iteratively grabs two scans using the ScanGrabbing procedure and register them using the pIC algorithm. It is worth noting that the pIC takes as input two consecutive scans (S_{ref} and S_{new}) and its relative displacement which coincides with the position occupied by the robot at the end of the first scan ($\hat{\mathbf{q}}_{\mathbf{ref}}$). The output is an improved estimation of the robot displacement ($\hat{\mathbf{q}}_{\mathbf{new}}$). The iterative compounding of the relative displacement allows to track the global robot position.

Algorithm 1 MSISpIC

$MSISpIC()$ {
  $[S_{ref}, \hat{\mathbf{q}}_{\mathbf{ref}}, \mathbf{P}_{\mathbf{q_{ref}}}] = ScanGrabbing()$
  $\hat{\mathbf{q}}_{\mathbf{global}} = \mathbf{0}$
  while(**true**) {
    $[S_{new}, \hat{\mathbf{q}}_{\mathbf{new}}, \mathbf{P}_{\mathbf{q_{new}}}] = ScanGrabbing()$
    $\hat{\mathbf{q}}_{\mathbf{pIC}} = pIC(S_{ref}, S_{new}, \hat{\mathbf{q}}_{\mathbf{ref}}, \mathbf{P}_{\mathbf{q_{ref}}})$
    $\hat{\mathbf{q}}_{\mathbf{global}} = \hat{\mathbf{q}}_{\mathbf{global}} \oplus \hat{\mathbf{q}}_{\mathbf{pIC}}$
    $S_{ref} = S_{new};\ \hat{\mathbf{q}}_{\mathbf{ref}} = \hat{\mathbf{q}}_{\mathbf{new}}$
  }
}

2.1 Scan Grabbing

The scan grabbing procedure builds a static scan referenced to a single frame to be used by the pIC algorithm. Whenever a sonar beam is read and segmented using a predefined threshold, an Extended Kalman Filter (EKF) similar to the one used in Ribas et al. (2006) is used to estimate the robot's pose. The information of the system at step k is stored in the state vector $\mathbf{x}_k$ with estimated mean $\hat{\mathbf{x}}_k$ and covariance $\mathbf{P}_k$:

$$\hat{\mathbf{x}}_k = \left[\hat{\eta}^B, \hat{\nu}^R\right]^T \quad \mathbf{P}_k = E\left[\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)\left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)^T\right] \tag{1}$$

with:

$$\eta^B = [x, y, z, \phi, \theta, \psi]^T; \quad \nu^R = [u, v, w, p, q, r]^T \tag{2}$$

The vehicle's movement prediction is performed using the 6DOF kinematic model:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}) = \begin{bmatrix} \eta^B_k \\ \nu^R_k \end{bmatrix} = \begin{bmatrix} \eta^B_{k-1} + J(\eta^B_{k-1})\nu^R_{k-1}T \\ \nu^R_{k-1} \end{bmatrix} \tag{3}$$

$$J(\eta) = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi & 0 & 0 & 0 \\ s\psi c\theta & s\phi s\psi s\theta + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi & 0 & 0 & 0 \\ -s\theta & c\theta s\phi & c\theta c\phi & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & s\phi t\theta & c\phi t\theta \\ 0 & 0 & 0 & 0 & c\phi & -s\phi \\ 0 & 0 & 0 & 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \tag{4}$$

Although in this model the velocity is considered to be constant, in order to allow for slight changes, a velocity perturbation modeled as the integral of a stationary white noise v_k is introduced. The covariance matrix $\mathbf{Q_k}$ of this acceleration noise is diagonal and in the order of magnitude of the maximum acceleration increment that the robot may experience over a sample period.

$$\nu^R_k = \hat{\nu}^R_k + v_k T \tag{5}$$

$$E[v_k] = 0; \quad E[v_k v_j^T] = \delta_{k_j} \mathbf{Q_k} \tag{6}$$

Hence, v_k is an acceleration noise which is first integrated and then added in velocity (5), being nonlinearly propagated to the position.

Since we are only interested in the robot's relative position (and uncertainty) with respect to the beginning of