An Implementation of a Behavior based Control Architecture for an Autonomous Underwater Vehicle

Arsalan Malik[1], Muhammad Naveed[1] and Wajahat Kazmi[1]
Escola Politècnica Superior, Universitat de Girona, Girona, Spain, 17004
{u1066650,u1066663,u1066662}@correo.udg.es

***Abstract* — In this paper an implementation of a behavior based control architecture for an Autonomous Underwater Vehicle (AUV) is presented. The control architecture attempts to plan a path for AUV's navigation in a 3D environment with unknown obstacles. In addition, it maneuvers the AUV to reach its goal while avoiding obstacles. The behavior based control architecture implemented is based on Bug algorithm.**

I. Introduction

NAVIGATION is an important problem for autonomous robots. Given partial knowledge of its environment and series of static goal positions, navigation involves the ability of an autonomous robot to act based on its knowledge and sensor information so as to reach its goal(s) as efficiently and as reliably as possible.

This paper details the work performed for a path planning simulation of an autonomous underwater vehicle (AUV) operating in a 3D environment with obstacles. The objective is to find a collision-free path from a given initial position to known goal points. A complete path planning algorithm should guarantee that the robot can reach the target if possible, or prove that the target cannot be reached. The choice of path is a crucial performance consideration, for example, a smooth path is desirable for practical robots.

Our mission planning algorithm is based on Bug Algorithms proposed in [5] and detailed in [10]. First, related work is reviewed in section II. The environment model is presented in section III. The Bug Algorithms proposed by [5] are briefly reviewed in section IV. The design of our approach is given in section V and finally results are presented along with discussion.

II. Related Work

A number of approaches have been proposed by the researchers in this regard. In realistic world, the autonomous robot does not have prior knowledge about its environment, hence it must use its sensors to perceive environment and plan its mission. There are two types of sensor based approaches. In global sensor based approach, the robot builds a global map based on sensory information and uses it for mission planning [1]-[3]. This approach guarantees that either the target will be reached or the robot will map its entire accessible map and conclude that the target is unreachable. In contrast, local sensor based approach uses local sensory information in a purely reactive fashion. In every control cycle the robot uses its sensors to locate obstacle in the path and it plans its next action. Many researchers have addressed this problem. Different Bug algorithms have been proposed [4]-[6]. They are complete (the solution will be found if it exists, otherwise the algorithm indicates that there is no solution), but in terms of smoothness their performance is not satisfactory: when the robot switches between different segments of the path it moves abruptly. An alternative to these algorithms is behavior based control architecture [7] and [8]. The drawback of this approach is that the performance is difficult to quantify formally. In [9], a hybrid control approach is used to combine behaviors for mobile robots.

III. Environment Model

This simulation uses an environment in which the robot is operating in a 3D manifold. It contains a finite number of static obstacles. Obstacles' contours are assumed to be reasonably smooth curves. It has one starting point S and three goal positions G1, G2 and G3. The robot is considered to be a point. During entire mission the coordinates of robot's current position C and next goal G (1, 2 or 3) are known. The robot is equipped with eight limited range sonar sensors, which provide readings from 0-10m at 0, 45, 90, 180, 270 and 315 degrees, upward and downward. The robot can measure the distance to the obstacles which are within the sensors' range. The robot's translational (X±), vertical (Z±) and rotational (YAW±) velocities can be controlled within a range [-1, +1]. This setup corresponds to a path planning problem with incomplete information about the environment as robot does not know the location and shape of the obstacles until they are within the sensors' range. Local sensory information is used for feedback control of the robot's motion. This model is attractive because many practical robots operate in unknown and changing environments. A complete map of the environment along with its parameters is given in figure 1.

1. Erasmus student of European Masters in Computer Vision and Robotics (VIBOT)

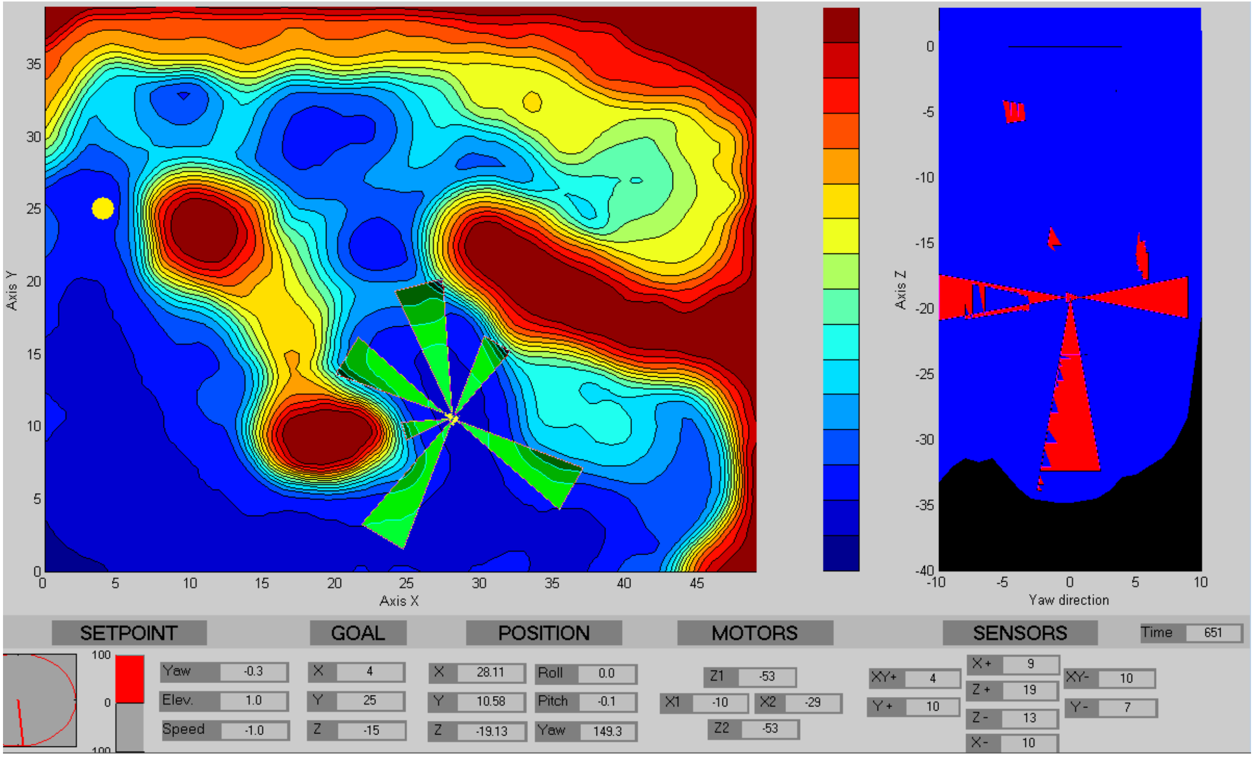

Fig. 1. Robot's complete simulated environment along with its parameters

IV. BUG ALGORITHMS

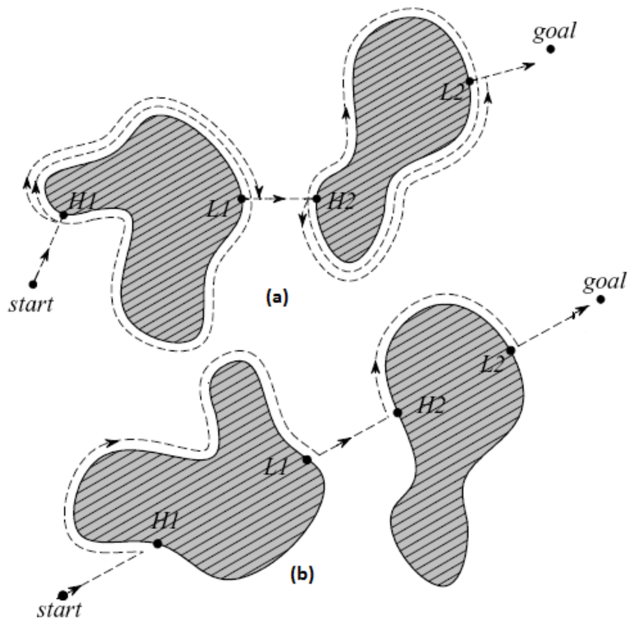

Fig.3. Bug Algorithms: (a) Bug-1 and (b) Bug-2

The Bug algorithms as proposed by [5] are perhaps the simplest obstacle avoidance algorithm one could imagine. They require only current sensor readings and approximate information regarding the direction of the goal to plan robot path. The basic idea is to follow the contour of each obstacle in the robot's way and thus circumnavigate it. Two slightly different variants of Bug algorithm are [10]:

- Bug-1: Robot fully circles the obstacle first and then departs from the point with the shortest distance toward the goal.
- Bug-2: Robot follows the obstacle's contour, but departs immediately when it is able to move directly toward the goal.

Figure 2 shows these two approaches. In Bug-1, the approach seems very inefficient but guarantees that the robot will reach any reachable goal. Bug-2, on the other hand is an improved Bug algorithm having significantly shorter total robot travel.

V. OUR APPROACH

The main ideas of Wall Following and Goal Seek behaviors of Bug-2 algorithm as suggested by [10] are implemented. Following this concept, behavior based control architecture is designed in which two different behaviors are coordinated through a single controller. The mission planning of the robot is a function of three parameters, i.e. its sensor values, current position and orientation of the robot and position distance of the goal.

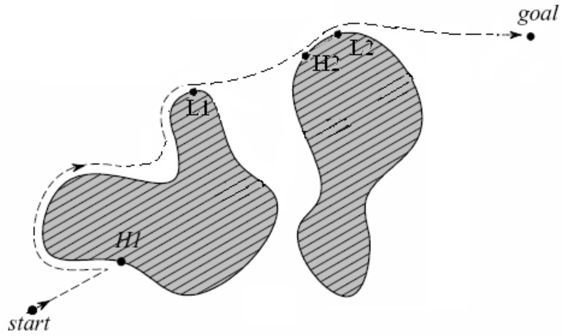

Fig.3. Trajectory of robot following Goal Seek and Wall Following (Obstacle Avoidance) behaviors based on Bug Algorithm

A. *Goal Seek Behavior*

The Goal Seek behavior is governed by the position vector to the goal from robot's current position and robot's current orientation. This behavior aims to minimize the distance between robot and goal, by computing the required Yaw angle and forward velocity of the robot. Once it is detected that robot is in proximity of goal, the behavior adjusts downward and forward velocity of the robot. The pseudo code of this behavior is given below:

1. Compute Position Vector of the Goal
2. Compute required Yaw angle of the robot
3. If distance to Goal is greater than a threshold, set Yaw and Forward velocity of the robot
4. Else, set Yaw, Forward and Downward velocity of the robot

B. *Wall Following Behavior*

The Wall Following behavior essentially attempts avoid collision by escaping from local minimum of the goal position vector. This is achieved by following wall (contour) if an obstacle is detected between robot's current position and the goal. This is illustrated by figure 3. When robot reaches position H1 following Goal Seek behavior, it detects an obstacle. It than follows Wall Following behavior along the contour of the obstacle, and reaches L1. At position L1, it detects that there is no obstacle between robot and goal using its sensors, thus activates Goal Seek behavior. This process is repeated again at H2. This behavior detects obstacle in every control cycle. If no obstacle is detected, the behavior informs coordinator to activate Goal Seek behavior. The pseudo code of the Wall Following behavior is given below.

1. Detect obstacle using on board sonar range finder
2. If obstacle is detected, determine avoidance act based on position of the obstacle and the attitude of robot
3. Set Yaw angle and Forward velocity based on avoidance act
4. If no obstacle is detected, inform coordinator to activate Goal Seek behavior

The parameters such as goal and sensor thresholds and robot velocities are optimized for the given environment settings.

VI. EXPERIMENTS

The behaviors implemented were tested in our simulated environment described before. The trajectory of the robot in our test environment is given in figure 4.

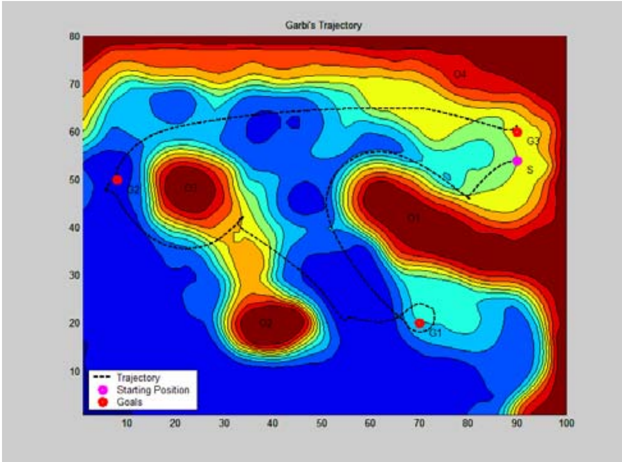

Fig. 4. Trajectory of the robot in our simulated environment

The robot first followed Goal Seek behavior until it detected first obstacle O1. It followed the contour of the O1, until it found no obstacle. The robot reached G1 following Goal Seek behavior. From G1, it started following Goal Seek behavior, until it detected obstacle O2. Briefly following the contour of this obstacle, it followed Goal Seek behavior until it encountered O3. After following the contour of O3, it reached G2 following Goal Seek behavior. From G2, the robot first followed Wall Following behavior, than Goal Seek behavior until it detected O4. After following Wall Following behavior, it finally reached G3. The first goal was achieved in approximately 600 time units, second goal in 400 time units and third goal in 500 time units.

VII. CONCLUSION

We have shown that a simplified form of Bug-2 algorithm can be efficiently used for the task of seeking goal while avoiding obstacles. It can be adapted to any environment with unknown obstacles by changing the parameters.

ACKNOWLEDGMENT

The authors would like to thank Marc Carreras and Pere Ridao for providing the MATLAB simulation tool.

REFERENCES

[1] G. Fouxm M. Heymann, and A. Bruckstein, Two dimensional robot navigation among unknown stationary polygonal obstacles, IEEE Transactions on Robotics and Automation, 9(1):96-102, 1993

[2] A. Stentz, Optimal and efficient path planning for partially known environments. In Proceedings of the IEEE Conference on Robotics and Automation, pp3310-3317, 1994

[3] A. Zelinsky, Using path transforms to guide the search for findpath in 2D. The International Journal of Robotics Research

[4] V. J. Lumelsky and A. A. Stepanov, “Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape,” Algorithmica (New York), vol. 2, no. 4, pp. 403–430, 1987.

[5] Y.H. Liu and S. Arimoto. Path planning using a tangent graph for mobile robots among polygonal and curved obstacles. International Journal of Robotic Research, 11(4):376-382, 1992.

[6] I. Kamon, E. Rivlin, and E. Rimon, A new range-sensor based globally convergent navigation algorithm for mobile robots, ser. Proceedings - IEEE International Conference on Robotics and Automation. Minneapolis, MN, USA: IEEE, Piscataway, NJ, USA, 1996, vol. 1, pp. 429–435.

[7] R. C. Arkin, Behavior-based robotics. Cambridge, MA: MIT Press, 1998.

[8] D. Kortenkamp, R. P. Bonasso, and R. Murphy, Eds., Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems. Cambridge, MA: MIT Press, 1998.

[9] M. Egerstedt and X. Hu, “A hybrid control approach to action coordination for mobile robots,” Automatica, vol. 38, no. 1, pp. 125–130, 2002.

[10] Roland Siegwart and Illah R. Nourbakhsh, Introduction to Autonomous Mobile Robots, The MIT Press, 2004