Measurement 73 (2015) 68–73

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

Development of an obstacle-avoidance algorithm for snake-like robots

Şahin Yıldırım [a,*], Ebubekir Yaşar [b]

[a] Mechatronic Engineering Department, Engineering Faculty, Erciyes University, Kayseri, Turkey
[b] Vocational School, Computer Programming Dept., Gaziosmanpaşa University, Tokat, Turkey

ARTICLE INFO

Article history:
Received 24 December 2014
Received in revised form 2 March 2015
Accepted 27 April 2015
Available online 5 May 2015

Keywords:
Avoiding obstacles
Path planning
Snake-like robot

ABSTRACT

In this work a path planning which is the first step of motion planning in especially snake-like robot applications, away from an obstacle in an environment where exist many obstacles is developed. Different from the algorithms in literature, a path away from an obstacle is planned without determining the configuration free space in a place that contains many different shaped obstacles with the help of intelligent objects that are created object oriented programming (OOP) techniques. With the help of this developed algorithm not only the probable paths but also finding the shortest, safest and fastest path and correcting it with the help of intelligent objects are evaluated at the same time. Also user can adjust relationship between performance and path quality by changing object count.

1. Introduction

The motion planning process always produces motion in order to combine starting point with the target point avoiding obstacles. A robot performs tasks by executing motions in the workspace [1]. Robot motion planning focuses on only robot changes and necessary turns undervaluing the dynamic structures and limitations [2].

Path planning works are based on finding a path away from an obstacle. In this situation researchers are focused on producing an algorithm which provides a path without any crash. Planning a path is highly needed for mobile robots in accordance with the concept that a robot can make its duties with the help of motion in real world. Many methods are developed in this subject.

Motion planning are searched from many researches during 1970s. For the first time in 1969, Nilsson made a mobile robot definition which had the ability of motion planning. Nilsson introduced the visibility graph method which was combined with A^* searching algorithm, that provides a point robot find the shortest way away from obstacles within the polygonal and geometrical shaped obstacles. This method remained very popular [3].

Udupa [4] suggested the idea of making the robot as small as a point for avoiding obstacle algorithms. With the help of this idea Lozano Perez and Wesley suggested a more general and systematic idea in order to plan a path for polyhedral/polygonal robots without touching polyhedral/polygonal obstacles [5]. This work pave the way for the idea of configuration space. This study led to the idea of configuration space. This configuration idea expresses the robot to be represent just one point in parameter space that defines robot's degrees of freedom [3].

While the representing forbidden areas in configuration space, the opposite part of configuration space represents free space [3]. This is the probable motion area of the robot. In 1979 Lozano-Perez and Wesley investigated the shortest way in their works changing the method named visibility

* Corresponding author.
E-mail addresses: smakyuk@hotmail.com (Ş. Yıldırım), ebubekir.yasar@gop.edu.tr (E. Yaşar).

http://dx.doi.org/10.1016/j.measurement.2015.04.021
0263-2241/© 2015 Elsevier Ltd. All rights reserved.

diagram into graph theory thanks to line which is formed combining the whole obstacles' face to face corners [5,6].

Andrews and Hogan in 1983 and Khatip [7] in 1985 developed the potential field method by thinking the area, where the robot is working is in the imaginary potential field. In this potential field where the robot is in, target point forms attractive and obstacles form a propulsive potential field. The robot in this potential field goes toward to the target avoiding the obstacles effected by attractive and propulsive forces as if there is a slope in the field. Having local minimums is the biggest problem of this potential field method. This situation brings out the result that as if the robot reaches the target before it reaches the target and this situation is named local minimum.

In 1987 Chazelle [8] investigated the path that goes the target over the cells that did not touch the obstacles with a technique he developed by dividing the working area, where the obstacles also exist in, into approximate decomposition cells. If the cells touching the obstacles does not reach the target, the action of dividing is repeated by making the cells smaller. One of different application of cell algorithm is vertical decomposition method. Here, center point of the line connected between the two obstacles or obstacles and any corners is obstacles-free points. The shortest path is investigated changing these points into graph theory like in visibility diagram method [2].

In 1991 Aurenhammer's voronoi diagram method divides the plane according to nearest neighbor rule [9]. This rule is each point is related to nearest plane field. The diagram is formed combining the points that exist at the same distance to its two nearest obstacles. The shortest path is investigated among the probable paths with this diagram.

In 1994 Kavraki et al. [10] tried to combine the near corners with a line help of local planner accepting the point, which is randomly taken from configuration space as a corner point if the point belongs to free space (C_{free}) in their probability based path finding method (PRM). Local planner controls the formed line if they are on the obstacles or not. Valid combinations are added to graph theory.

LaValle and Knuffer [11] in 1998 formed tree structures finding and connecting the nearest points and step by step widening the first example which is taken from configuration space in their algorithm. That they developed in their random three structured fast searching method (RRT). In this three structure new points exist randomly. The field is formed from three structure which had connecting branches as many as the repetitions in this way.

2. Intelligent object algorithm

In this work a direct-line connection is created between the starting point and the target by connecting the intelligent objects each other which are produced with Delphi one of object oriented programming language. It is not needed also to determine the free configuration space that reduces the robot as small as a point. Because if the size of the objects are determined according to the maneuver ability of the robot and time spent, the configuration space is not needed. In this work the environment is accepted being two dimensions and the obstacles randomly may have any shape, position and edge number.

2.1. Structure of intelligent object

To solve the obstacles avoiding problem we used OOP based objects. An objects has properties, events and methods is element of the program. An object that has properties, events and methods is a small part of the program. An object stores its state in fields (variables in some programming languages) and exposes its behavior through methods (functions in some programming languages) [12]. OOP is a programming paradigm that represents the concept of "objects" that have data fields (attributes that describe the object) and associated procedures known as methods. Objects, which are usually instances of classes, are used to interact with one another each another to design applications and computer programs [13,14]. An object has both state (data) and behavior (code).

2.1.1. Object properties

Determines the features of the objects. Fields and properties represent information that an object contains. Fields are like variables because they can be read or set directly. Properties have get and set procedures, which provide more control on how values are set or returned [15]. In Fig. 1 some properties of the intelligent objects are shown.

Width, Height: Size of the objects.
Color: Color of the objects.
X, Y: Coordinates of the objects
Image: Image that has obstacles
Header: First and last objects cannot move during running algorithm.
Next: Neighbor object.
Dist_O: Distance between obstacles and objects.
Shape: Shape of the object.
Speed: Used finding the fastest path. Each object has speed parameter according to its size and coefficient of friction. This properties used especially finding path for snake robots. As known snake robots can move faster in wide areas.

2.1.2. Object events

Events enable a class or object to notify other classes or objects when something of interest occurs [15]. Events alert applications when there is a change of state.

OnCreate: Overridden standard Windows Create event.
OnContact: It is fired when objects are above any obstacles.

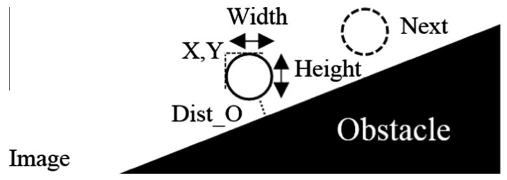

Fig. 1. Properties of the intelligent object.