Co-Design of Reactive Embedded System for Motion Control in Hostile Environment

Narayanan Subramanian, Sunil Pandita & Rabi Mahapatra

Department of Computer Science
Texas A&M University
College Station, TX 77843-3112
{narayann,spandita,rabi}@cs.tamu.edu

Contact Author: Rabi Mahapatra (rabi@cs.tamu.edu)

• What is Original contribution of this work?
In this paper we have proposed an optimized automated path-planning algorithm for moving objects in a hostile environment. Hardware-Software Codesign method has been adopted to implement the algorithm on embedded system so as to control the motion of the moving objects and determine the path in demand. An efficient partitioning strategy has been deployed to reduce the design cost and power requirements while achieving the high performance.

• Why should this contribution be considered important?
The importance of this work lies in the variety of applications in which this system can be used. Some of these applications are 1) Missile avoidance systems – used by an aircraft to move over enemy territory avoiding all the missiles. 2) Automated vehicular traffic system – unmanned vehicular systems. 3) Automated office robots – used to design robots that can move around transferring files etc. in an office. Besides, the approach considered in this paper is flexible with short design turn around time.
ABSTRACT

The various contemporary path-planning algorithms can be broadly classified into two main paradigms, namely Local Path Planners (LPP), e.g. Dynamic Window Approach and Global Path Planners (GPP). However there are inherent shortcomings in each of these aforementioned approaches. The LPP suffer from the drawback that they may get caught in local optima, and thus may not deliver the globally optimal solution. On the other hand, the GPP approaches provide the optimal solution but do not guarantee that the obstacles shall be avoided. In this paper we propose an algorithm that combines the positive aspects of both these approaches, to find the globally optimal solution as well as avoid collisions with the obstacles. The system considered consists of an object moving from a start point to a goal point, avoiding collisions with obstacles in its path, which might be stationary or moving in any arbitrary fashion.

The heart of the proposed algorithm is the heuristic function that comprises of two parts: \textit{Angle Correction} and \textit{Find Nearest Obstacle}. The shortest distance from the start point to the goal point is the straight-line path between the two points. But when there are random moving obstacles in the path, it gives rise to a complex scenario. From any position, the shortest distance is still the line joining the current position to the target point. So when we minimize the angle between the current velocity and the line joining the current position to the target point, the resultant path is the shortest one. Also, we take into account of collision avoidance by keeping track of closest obstacles at any instant. Theoretically, the justification of the algorithm can be explained with the help of the Figure 1.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.png}
\caption{The Subset Explanation}
\end{figure}

Out of the total set of possible velocities in the next interval, only a few cases are possible because of limited acceleration, a physical constraint for the objects. Therefore, in the above figure, instead of searching for the entire two-dimensional space we have to search only the rectangular part. Further, with in the rectangle, we reduce the search space by eliminating the velocities that lead to a collision. So effectively we search for only a small subset of the overall search space.

The main functional blocks of the system are \textit{Controller, Heading, Heuristic Function, Emitter, Sensor, Find Nearest Obstacle and Motion Predictor}. Figure 2 illustrates the interaction between these modules.

In this paper we proposed a hardware-software partitioning of the above system. The system consists of heterogeneous components with different timing requirements. The factors that were considered for partitioning the system are Time Criticality, Re-programmability, Parallelism, Inter-Module Communication, Performance Bottlenecks, Chip Area, Power Dissipation.
A comparison of the performance of the dynamic window approach and our approach indicates that the dynamic window approach is comparable only for a few obstacles is (four or less). However, as the number of obstacles increase, the performance of the proposed approach significantly excels. On the other hand the GPP approach uses the least number of computational steps to get to the destination but it does not guarantee collision Avoidance.

In our approach, the time taken for the object to traverse a given distance depends on the number of obstacles it encounters, as well as their relative positions and trajectories. This is because, every time an obstacle is encountered the object detours from the shortest path to the goal, therefore resulting in more iterations of the objective function for calculating the shortest path. The Codesign strategy can achieve the performance requirements by adopting suitable trade-offs among cost, delay and power consumption.