2D Path Planning Based on Dijkstra's Algorithm and Pseudo Priority Queues

Jose Guivant, Brett Seton and Mark Whitty
School of Mechanical and Manufacturing Engineering, The University of New South Wales, Sydney, Australia
j.guivant@unsw.edu.au

Abstract

This paper presents the application of the PPQ Dijkstra approach for solving 2D path planning problems. The approach is a Dijkstra process whose priority queue (PQ) is implemented through a Pseudo Priority Queue (PPQ) also known as Untidy PQ. The performance of the optimization process is dramatically improved by the application of the PPQ. This modification can be used for a family of problems. The path planning problem belongs to the family of feasible problems that can be solved by considering PPQ-Dijkstra approach. The solution provided by the PPQ-Dijkstra algorithm is optimal, i.e. it is identical to the solution obtained through the standard Dijkstra algorithm. The PPQ-Dijkstra algorithm can be also applied for higher dimensionality problems such as non-holonomic planning processes, e.g. involving configuration spaces of higher dimension.

Feasible Problems

The PPQ-Dijkstra algorithm can be used in problems where the cost of transition between states (nodes or edges), $\delta(l,k)$, has a non zero lower bound, $\delta_{\min}$. This means that there is no possible transition between any couple of states (l,k) having a cost $\delta(l,k)$ lower than $\delta_{\min}$.

$$\delta(l,k) \geq \delta_{\min} \quad \forall (l,k)$$
$$\delta_{\min} > 0$$

If that condition is satisfied then the standard Priority Queue (PQ) can be replaced by a Pseudo Priority Queue (PPQ). A PPQ is a less strict version of a PQ, and its computational cost is lower than the usual PQ. The implication of this fact is that the cost of performing a PPQ-Dijkstra is proportional to the number of visited sates.

The condition means that the total cost for reaching any 1-step reachable state would not be less than the current state value plus $\delta_{\min}$. Consequently it is a sufficient condition for the Dijkstra process to operate optimally if the PQ at least maintains an internal order where only if the cost of the state ***i*** is lower than the cost of state ***k*** minus $\delta_{\min}$ then the state ***i*** would have more priority than state ***k***. i.e. the state ***i*** must be located before the state ***k*** in the queue,

$$G_C(i) < G_C(k) - \delta_{\min} \quad \Rightarrow \quad prio(i) > prio(k)$$

where the operator $G_C(k)$ means current global cost (e.g. cost-to-go in backward planning) of state ***k*** and $prio(k)$ means its associated priority in the queuing stage.

This condition is less demanding than the strict condition

$$G_C(i) < G_C(k) \quad \Rightarrow \quad prio(i) > prio(k)$$

that governs the standard PQ ordering scheme.

This means that if two states (i,k) meet the condition $\left|GC(i) - GC(k)\right| < \delta_{\min}$ then their relative priorities are irrelevant.

This fact is exploited in order to reduce the processing requirements for maintaining the Priority Queue consequently improving the real-time capabilities of the planner.

References

Dijkstra, E.W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1(1):269-271.

Sethian, J. A. 1996. A fast marching level set method for monotonically advancing fronts. In *Proceedings of the National Academy of Sciences of the United States of America*, 93(4):1591-1595.

Yatziv, L., Bartesaghi, A., Sapiro, G. 2006. O(N) implementation of the fast marching algorithm. *Journal of Computational Physics* 212(2):393-399.

Robledo, A., Cossell, S., and Guivant, J. E. 2010. Pseudo priority queues for Real-Time performance on dynamic programming processes applied to path planning. In Proceedings of the 2010 Australasian Conference on Robotics & Automation, Brisbane, Australia.