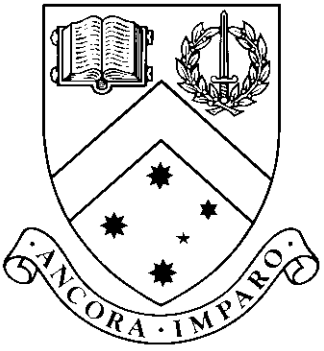
School of Computer Science
Monash University



Research Proposal — Comp Sci Honours, 2017

Exploring improvements to Part-to-picker systems in Warehouses

Phillip Wong 25150510

Supervisor: Daniel Harabor

Contents

Abstract

The order picking process is the number one expense when looking at the operational costs in warehouses. This project will look at *part-to-picker*, a method of order picking where products are autonomously retrieved and delivered to the picking areas. Previous research has improved on multi-agent path finding algorithms (MAPF) but mostly overlooked other aspects. Here we will be exploring the effects of warehouse layout. In particular, this involves investigating a number of modifications which are expected to determine how we decide a good warehouse layout. These include: introducing an intermediate dropping zone, adding the capability for robots to maneuver under storage pods and optimizing order processing. The results of this project will help identify how we should layout storage and picking stations in a warehouse. Additionally we will be looking at developing a MAPF method which uses a pre-computed path oracle.

1 Introduction

Order picking is a process in warehouse systems whereby a product is retrieved according an incoming customer order. This process has been identified by De Koster et al. (2007) as the most expensive process in operating a warehouse, estimated to take 55% of the warehouse operating cost.

Here we look at a method of order picking known as part-to-picker systems which automates the movement of products from where they are stored to picking stations, where workers will manually pick and process the product. Part-to-picker systems often employ the use of automated vehicles to retrieve and deliver orders from where they are stored. AutoStore (2015) is a recent system where products are organized in a grid of stacked bins. Robots move around the top of the grid, lifting bins and delivering them to a human picker. Benefits of the AutoStore system include high storage density and expansion capability. While not much literature is published about the specifics of AutoStore, we suspect the major downsides are: slow, expensive order retrieval as well as high infrastructure and maintenance costs.

In this project, we look at Kiva Systems (now known as Amazon Robotics). In Kiva systems, products are stored in shelves known as storage pods. Robots known as drive units are responsible for retrieving and delivering storage pods to picking stations before returning them to an appropriate place in the warehouse. A human worker is stationed at each picking station who picks the item off the pod before processing it (Fig 1).

Figure 1: A worker picking an order from a storage pod. The orange robot underneath is the drive unit. (Al Dekin (2014))

Kiva systems do not require a complex infrastructure to operate, a warehouse needs only storage pods, a picking station and a suitable number of drive units to operate. As long as the warehouse has space, more robots, pods or stations can be easily be added to the system to satisfy the incoming flow of customer orders. When a drive unit malfunctions it can be easily accessed and replaced. In summary, the main benefits of Kiva systems are their low initial and maintenance costs, rapidly deployment and flexibility (Wurman et al. (2008)).

1.1 Research questions

In order to explore the improvements that can be made to improving Kiva systems, we aim to answer two main questions:

1. What effect will these modifications have and how do they affect our decision when it comes to organizing the warehouse layout?
    - An optimized order process
    - Allowing drive units are capable of maneuvering underneath storage pods
    - An intermediate zone for drive units to drop off storage pods
2. How much faster will the MAPF search run by pre-computing paths and storing them in a path oracle?

2 Background

In Kiva Systems we face a multi-agent pathfinding (MAPF) problem. MAPF aims to find a path for each agent so that they can reach their goal while ensuring that no path conflicts with another. When analyzing the performance of a MAPF solution we generally aim to reduce the makespan of the system. Specific to Kiva systems we also aim to reduce the downtime of picking stations. Outside of order picking and warehouse applications, MAPF has usage in video games, robotics (Bennewitz et al. (2002)), search and rescue (Konolige et al. (2006)) and automated ports.

Finding an optimal solution to multi-agent pathfinding is an NP-hard problem (Yu and LaValle (2013)) and has found applications in systems containing a small number agents. This is not an option as Kiva systems deal with hundreds of agents, for example the Office Supply company, Staples uses 500 robots in their $30000m^2$ center (Guizzo (2008)). The best we can get currently is a bounded suboptimal solution and this has been explored by a number of existing literature (Cohen and Koenig (2016)).

Generally methods are provided to simplify the MAPF problem, Cohen and Koenig (2016) uses user-provided highways to help guide agents towards a specific direction, greatly reducing the chance of path collisions. Wilt and Botea (2014) identifies bottlenecks in the environment and assigns a controller to handle agents who want to pass through this area, simplifying behaviour in high collision zones. Another common technique is grouping agents into teams. Ma and Koenig (2016) splits agents into teams of 5 and presents a Conflict-Based Min-Cost-Flow algorithm which and shows that they can achieve a correct, complete and optimal solution.

Specific to order picking, it is possible to assist MAPF by manipulating incoming orders to suit the location of products. For example if a large order of one product came in, this could be interlaced with other orders, so agents do not need to all head to the same location. In a similar fashion, MAPF can be assisted by smart distribution of products across the warehouse. Take the same example where a large order of one product comes in, if the products were spread out around the warehouse this would not be an issue. This method is especially relevant in Kiva systems as storage pods can be easily moved around

the warehouse by the drive units. Boysen et al. (2017) looks at both of these aspects and found that with optimized order processing, only half the units are required to provide the supply given by a non-optimized system.

3 Methodology

3.1 Path oracle

Decentralised approaches are usually used in MAPF algorithms due to their lower CPU and memory requirements (Wang et al. (2009)) compared to a centralised approach. Decentralised MAPF involves agents independently searching for a path to their goal. Here we aim to introduce a path oracle which will pre-compute paths, removing the need for agents to perform a search at runtime. We expect to base this off the work by Strasser et al. (2015), utilizing a Compressed Path Database.

Once a path is found, agents take turns following their path one step at a time until a spatio-temporal conflict occurs between agents. We choose one agent based on a user-defined utility function and this agent will continue along the path. The other agent has two options first is simply waiting until the other agent moves out of the way. The second method we may explore involves using a looking forward using a spatio-temporal reservation table to find a path without conflicts (Wilt and Botea (2014)).

3.2 Warehouse layout

Usually the picking station is positioned on one side of the warehouse and the pods are laid out in rows (Fig. 2).

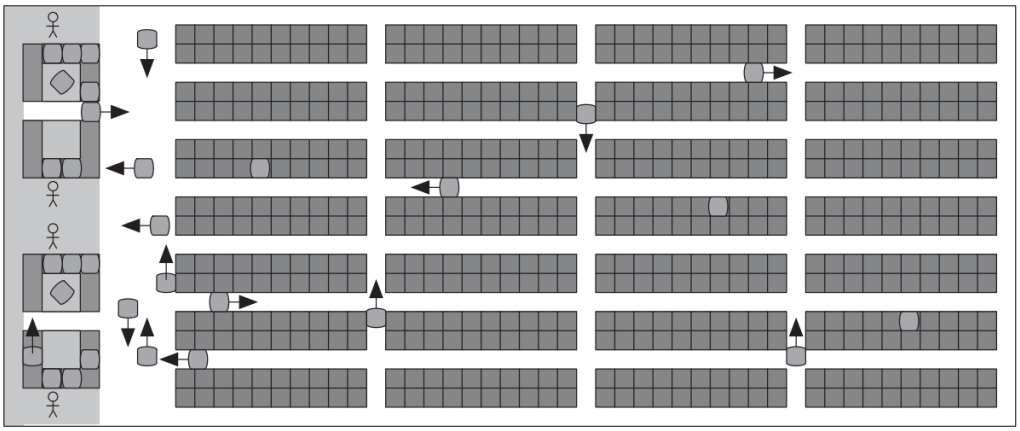


Figure 2: A Small Region of a Kiva Layout (Wurman et al. (2008)). Picking stations located on the left and storage pods laid out in rows.

One of the focuses of this study is designing logical warehouse layouts, where picking stations may be distributed inbetween or around picking stations. We want to look at how this affects makespan and may help or cause issues in the MAPF. Ultimately to identify what works and what does not. A hand-made set of warehouse layouts will be the start but one possibility is the use of genetic algorithms to automatically generate a good warehouse layout.

In Section 2, we saw that Wilt and Botea (2014) identified bottlenecks in the environment and assigned controllers to simply behaviour for any agents who wanted to travel through that zone. Inspired by this, we plan to split the warehouse into two halves and introduce an intermediate zone (See Fig 3). Now, drive units in the far zone deliver pods to the intermediate zone instead of a pickup station. Drive units situated in the delivery

zone will be responsible for retrieving pods in the intermediate zone and bringing them to pickup areas. These zones will have their own controller which handles any agents within the zone and tells agents what behaviour should occur.

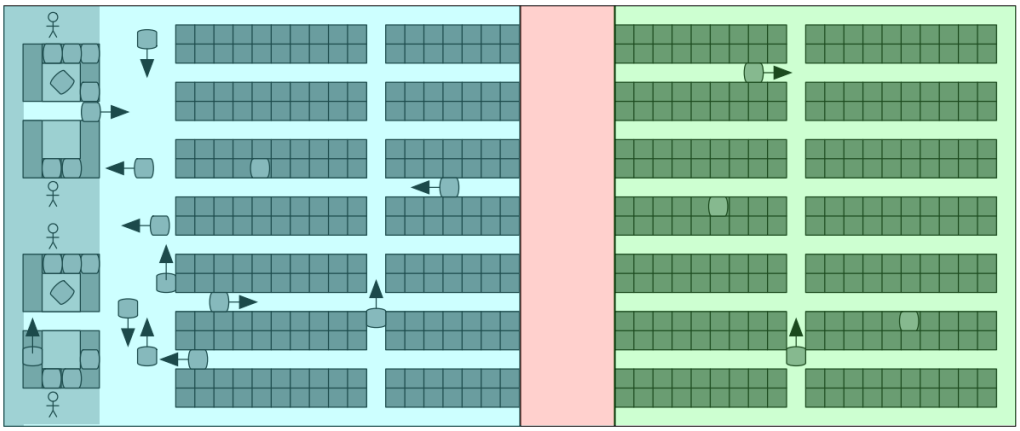


Figure 3: Intermediate zone in red, delivery zone in blue and far zone in green

3.3 Order processing

This process is described in detail in Section 2. Reiterating the main points of order processing we look at distribution of products around the warehouse to reduce path collisions and smart sequencing of incoming orders according to the current distribution of products around the warehouse. This is important to consider as we expect that an optimized order processing sequence will decrease the effects of having an inefficient warehouse layout. Here we may take inspiration from Robin-hood hashing.

3.4 Capability for pathing below storage pods

One of the specifications for creating a drive unit is the ability to maneuver underneath a pods to pick it up, this means that with small adjustments to the dimensions of storage pods it is possible to allow drive units to maneuver underneath the pods. Until a unit starts carrying a pod, it will not see any obstacles in the environment besides other drive units. We will explore exactly how much effect this has on reducing the makespan of the system as it will definitely reduce the number of path collisions.

3.5 Timetable

There are a large number of individual tasks involved in this project which have been spread across the two semesters. I have identified two major tasks which are: implementing an optimized order processing method and implementing path oracle. These have been allocated multiple weeks in semester 2 and the path oracle has been prioritized over optimized order processing as we will have implemented robin-hood hashing earlier in week 11 of semester 1. Finally, some time has been allocated to presentations or deliverables related to this project and two weeks have been allocated to focus on university examinations.

Semester 1

Week(s)	Plan
7	Create warehouse simulation with simple A* pathfinding
8-9	Add multiple agents to the simulation using with Cooperative A* and moving between pickup stations
10	Implement an order sequencer assigning generating an inflow of products to be retrieved
11	Begin looking at robin-hood hashing
12-14	Focus on Interim Presentation, Literature Review and Examinations
Holidays	Look at adding an intermediate dropping zone (3.2) and implement ability to move under pods (3.4)

Semester 2

Week(s)	Plan
7-11	Implement Path Oracle with Compressed Path Databases
1-3	Implement an optimized order processing method (3.3)
5-7	Run simulation looking at combination of modifications and analyze results. Use any extra time to explore the use of genetic algorithms to generate warehouse layouts.
12	Finish first draft of Final Thesis
13-15	Focus on Final Presentation, Thesis and Examinations

4 Expected Outcomes

Overall we aim to contribute one major insight and one deliverable. In Section 3, we described methods to add: an optimized order process, the capability for drive units to path under storage pods and lastly addition of an intermediate zone. What we aim to answer is how these modifications may affect our decision when it comes to designing a good warehouse layout. To showcase our results we hope to produce two main graphs comparing the makespan of the system and the idle time of picking stations across our user-defined warehouse layouts. Accompanying this would be any interesting results we see from adding or removing modifications described in Section 3.

For our deliverable, we hope to produce an improved MAPF solution utilizing a precomputed path oracle and a simulation showcasing its usage in a simulated Kiva system. Similarly, we will showcase the results by comparing the makespan of the system and idle time of picking stations to alternative MAPF techniques.

References

Al Dekin, A. V. (2014). Kiva systems warehouse automation at quiet logistics.
URL: *https://youtu.be/3UxZDJ1HiPE?t=2m2s*

AutoStore (2015). Autostore introduction.
URL: *http://autostoresystem.com/thesystem*

Bennewitz, M., Burgard, W. and Thrun, S. (2002). Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots, *Robotics and autonomous systems* **41**(2): 89–99.

Boysen, N., Briskorn, D. and Emde, S. (2017). Parts-to-picker based order processing in a rack-moving mobile robots environment, *European Journal of Operational Research* .

Cohen, L. and Koenig, S. (2016). Bounded suboptimal multi-agent path finding using highways, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, AAAI Press, pp. 3978–3979.

De Koster, R., Le-Duc, T. and Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review, *European Journal of Operational Research* **182**(2): 481–501.

Guizzo, E. (2008). Three engineers, hundreds of robots, one warehouse, *IEEE spectrum* **45**(7): 26–34.

Konolige, K., Fox, D., Ortiz, C., Agno, A., Eriksen, M., Limketkai, B., Ko, J., Morisset, B., Schulz, D., Stewart, B. et al. (2006). Centibots: Very large scale distributed robotic teams, *Experimental Robotics IX*, Springer, pp. 131–140.

Ma, H. and Koenig, S. (2016). Optimal target assignment and path finding for teams of agents, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, International Foundation for Autonomous Agents and Multiagent Systems, pp. 1144–1152.

Strasser, B., Botea, A. and Harabor, D. (2015). Compressing optimal paths with run length encoding, *Journal of Artificial Intelligence Research* **54**: 593–629.

Wang, K.-H. C. et al. (2009). Bridging the gap between centralised and decentralised multi-agent pathfinding, *Proceedings of the 14th Annual AAAI/SIGART Doctoral Consortium (AAAI-DC 2009)* pp. 23–24.

Wilt, C. M. and Botea, A. (2014). Spatially distributed multiagent path planning., *ICAPS*.

Wurman, P. R., D'Andrea, R. and Mountz, M. (2008). Coordinating hundreds of cooperative, autonomous vehicles in warehouses, *AI magazine* **29**(1): 9.

Yu, J. and LaValle, S. M. (2013). Structure and intractability of optimal multi-robot path planning on graphs., *AAAI*.