2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Mathematical Modelling of Container Transfers for a Fleet of Autonomous Straddle Carriers

S. Yuan[1], B.T. Skinner[1], S. Huang[1], D.K. Liu[1], G. Dissanayake[1], H. Lau[2], D. Pagac[2], T. Pratley[2]
[1]*ARC Centre of Excellence for Autonomous Systems (CAS)*
University of Technology, Sydney, NSW, Australia
{shyuan, brad.skinner, sdhuang, dkliu, gdissa}@eng.uts.edu.au
[2]*Patrick Technology Systems, NSW, Australia*
{h.lau, d.pagac, t. pratley}@patrick.com.au

Abstract— **The main contribution of this paper is a mathematical model describing performance metrics for coordinating multiple mobile robots in a seaport container terminal. The scenario described here requires dealing with many difficult practical challenges such as the presence of multiple levels of container stacking and sequencing, variable container orientations, and vehicular dynamics that require finite acceleration and deceleration times. Furthermore, in contrast to the automatically guided vehicle planning problem in a manufacturing environment, the container carriers described here are free ranging. Although, the port structure imposes a set of "virtual" roadways along which the vehicles are allowed to travel, path planning is essential in preventing contention and collisions. A performance metric which minimises total yard-vehicle usage, while producing robust traffic plans by encouraging both early starting and finishing of jobs is presented for different vehicle fleet sizes and job allocation scenarios.**

I. Introduction

At seaport terminals, reducing the turnaround time of berthed ships and docked trucks in a highly uncertain and dynamic environment are the most significant factors in reducing the overall transportation cost of containers. A secondary, but competing objective is to minimise the utilisation of yard resources, in this case a fleet of *autonomous straddle carriers* (SCs). Any delays experienced while performing yard jobs using the SCs will impact the throughput of the entire seaport.

In this paper, we formulate a mathematical model which incorporates many of the challenging practicalities of an operational seaport container terminal. Including various nonlinear constraints related to the spatial and temporal aspects of SCs interacting with containers. Unlike many seaport terminals, which utilise yard vehicles that are either manned or fixed upon a circuit of tracks [1], this environment requires the use of a fleet of SCs. This creates the complex requirement of performing collision-free path planning for all transportations within the yard.

We have also proposed a sequencing parameter to account for multiple levels or stacks of containers stored in the yard. Stacking, creates an exclusion constraint for the set-down and pickup of containers, i.e. it is not possible to place a container on the upper tier without a container on the ground tier and it is not possible to pickup a container on the ground tier without initially removing a container present on the upper tier. Similarly, a sequencing parameter is proposed to solve the problem of loading and unloading of containers from trucks in a required order.

Another significant challenge is to address the orientation of containers as they are transported between the yard, ship(s) and truck(s). Since the orientation of containers placed onto ships and onto trucks is fixed in a single direction, path planning must consider the orientation at both the initial and destination nodes. In order to guarantee the required orientation prior to loading containers onto ships or trucks we model the changes in alignment during transportation (flip movement) as a container is transported from its initial node to destination node.

Furthermore, we have developed an objective function which considers the competing objectives of minimising the total SC utilisation while maximising the throughput of yard jobs. Here, the throughput of yard jobs is increased by encouraging the early starting and early finishing of jobs. Our approach does not introduce bias towards the allocation of short or long jobs, which can result in traffic plans that are more robust to uncertainties within the seaport. Once a method for determining optimal yard operations is established, prudent investment in capital and changes to the seaport infrastructure can be more accurately measured.

Apart from providing an excellent survey of the principle logistics processes and operations within current container terminals, the authors in [2, 3] suggest that future research in this area should incorporate "realistic situations" of the dynamic seaport container terminal operations into the models. It is in this spirit that our approach attempts to model several of the difficult practical challenges of a container terminal environment.

In a recent study, Preston and Kozan [4] examined mathematical modelling and optimisation using genetic algorithm and tabu search of the sea interfaces, specifically the transfer of export containers from the storage to berthed ships. Their implementation of the model and optimisation algorithms is capable of handling large problems that arise in the quay side operations. However, this study deals exclusively with import and export containers between the ship and the yard and does not include a comprehensive

978-1-4244-5040-4/10/$26.00 ©2010 IEEE

model that considers yard-to-yard and yard-truck transportations or other practicalities.

Also, Hartmann [5] proposed a general model for various scheduling problems that occur in container terminal logistics. The scheduling model consists of the assignment of jobs to resources and the temporal arrangement of the jobs subject to precedence constraints and sequence-dependent setup times. The model was applied to solve problems for SCs, AGVs, stacking cranes, and workers who handle reefer containers in the port of Hamburg. Good solutions where obtained using a resource constrained genetic algorithm.

The main contributions of this paper include the formulation of a mathematical model of a large and difficult multi-objective optimisation problem involving a fleet of fully autonomous SCs. The model formulation includes all container handling subsystems of an automated container terminal operating in Brisbane, Australia [6-8]. Additionally, a map of the yard environment and typical task lists have been developed as part of the simulation environment. Interested researchers will be able to use the model, map and list structures to conduct benchmarking of novel optimisation algorithms for simultaneous task allocation and path planning. The model will provide a basis for comparing solutions using an assortment of strategies. The rest of the paper is presented as follows. In Section II, we present the model formulation. In Section III, the simulation environment and a feasible solution are described. Finally, we discuss our future work and present conclusions in Section IV.

II. MATHEMATICAL MODEL

A. Overview of Container Terminal Environment

A yard environment map has been developed to model the actual Fisherman Islands Depot located within the Port of Brisbane, Australia. The yard environment map consists of 18380 positional nodes and 83155 predefined links. Currently, there are 23 autonomous SCs operating within the actual yard environment. The working area is strictly confined and SCs can travel freely from position to position, along predefined paths (links). Thus, the problem of optimising the assignment of tasks to SCs (task allocation) is complicated by the additional requirement to ensure safety through collision-free path planning, while attempting to meet the overall objective of minimizing the turnaround time of berthed ships and trucks docked at the truck import area.

Considering the requirements, let the map be represented by a graph (*N, L*), where all nodes are contained in a set *N* and all feasible links upon which identical SCs (*V*) can travel are contained in the set *L*. Nodes are not uniformly spaced within the map, hence links are not equal. All links in *L* are bi-directional and connect two neighbouring nodes. For each node in the map, there is a two-level stack where a container can be stored. That is, each node can be occupied by two containers vertically which adds significant complexity to the problem as both setdown and pickup sequencing must be considered.

Using a graph to represent the seaport map allows for the accurate determination of position and trajectory information at any time. As a result, both collision-free paths and optimal task allocation (online scheduling) can be calculated at any time for the purpose of the replanning problem.

B. Definitions of Parameters and Variables

This section further describes the model parameters and variables. Considering yard operations only, we can establish the following definitions.

Given Parameters:

V : set of identical SCs $\{V_1,...,V_{z_V}\}$, where V_i is a SC and $i \in \{1,...,z_V\}$, and $z_V = |V|$ is the total number of SCs.

B : set of containers (boxes) $\{B_1,...,B_{z_B}\}$, where B_i is a container and $i \in \{1,...,z_B\}$, and $z_B = |B|$ is the total number of containers. $B_i \in B : B_i = (\#)$.

N : set of nodes $\{n_1,...,n_{z_N}\}$, where n_i is a node and $i \in \{1,...,z_N\}$, and $z_N = |N|$ is nodes' total number.

L : set of map links, where l_{jk} is a link which connects two nodes (n_j, n_k), given by $\forall l_{jk} \in L : l_{jk} = \{n_j, n_k\}$.

$S_{n_i} \in \{0,1\}$: stack level (tier) at node n_i, where 0=ground level and 1=first level.

$\omega_{l_{ij}l_{jk}}$: predefined traversal data between every two adjacent links l_{ij} and l_{jk} : $(i,j,k) \in N$, $j \notin (i,k)$. Let $\omega_{l_{ij}l_{jk}} = 1$ if SCs need to decelerate and adjust direction and then accelerate for traversing from node n_i to node n_k via node n_j, otherwise, $\omega_{l_{ij}l_{jk}} = 0$.

J : set of jobs $\{J_1,...,J_{z_J}\}$, where J_k is a job and $k \in \{1,...,z_J\}$, and $z_J = |J|$ is the total number of jobs. Each job $J_k = \{B_i, InitNode, DestNode\}$ is associated with a container (B_i), an initial node ($InitNode = (n_I, S_{n_I}, AL_{S_{n_I}})$) and a destination node ($DestNode = (n_D, S_{n_D}, AL_{S_{n_D}})$). AL_{S_n} is container's door alignment. The specific definition is given in *Section C*.

$S_{lock}(p)$: set of locked nodes and links associated with the position p ($p \in L \cup N$) of SC (V_i) to prevent all collisions. Fundamentally, the requirement of collision-free path planning ensures that any locked nodes by a SC (V_i) cannot be occupied by other SCs at any time.

α_{ab} : picking job sequence. $\alpha_{ab} = \{0,1\} : \forall (J_a, J_b) \in J$. Let $\alpha_{ab} = 1$ if job (J_a) is to be picked before job (J_b), else $\alpha_{ab} = 0$.

β_{ab} : setdown job sequence. $\beta_{ab} = \{0,1\} : \forall (J_a, J_b) \in J$. Let $\beta_{ab} = 1$ if job (J_a) to be setdown before job (J_b), else $\beta_{ab} = 0$.

t_0 : starting time of a plan.

Δt_{pickup} : time required for a SC (V) to pick-up box.

$\Delta t_{setdown}$: time required for a SC (V) to set-down box.

$\Delta t_{accelerate}$: time for acceleration of SC (V) moving from zero to maximum (constant) velocity.

$\Delta t_{decelerate}$: time for deceleration of a SC(V) from its maximum (constant) velocity to zero velocity.

$MinTime^{J_k}$: the minimum theoretical processing time (from pick-up to set-down) for job (J_k).

TT_{xy} : intra-nodal travel time required between nodes $n_x : n_y$

Defined Variables:

$x_{V_i J_k}$: job assignment operator to map between a job (J_k) and a SC (V_i). Let $x_{V_i J_k} = 1$ if SC (V_i) is assigned to perform job (J_k), otherwise $x_{V_i J_k} = 0$.

$t^{J_k}_{start} / t^{J_k}_{finsh}$: planned start/finish time of job (J_k).

$t^{n_i}_{V_i\_arrive}$: planned arrival time at node (n_i) for $V_i \in V$.

$t^{n_i}_{V_i\_depart}$: planned departure time from node (n_i) for $V_i \in V$.

t_{idle} : total planned idle time of a SC.

$PT^{J_k}_{PUtoSD}(t)$: actual processing time for job (J_k) at a time (t).

$P^{V_i}(t)$: position of SCs ($V_i \in V$) at time (t), according to planned trajectory.

C. Definition of Container Orientation and Vehicle Trajectory

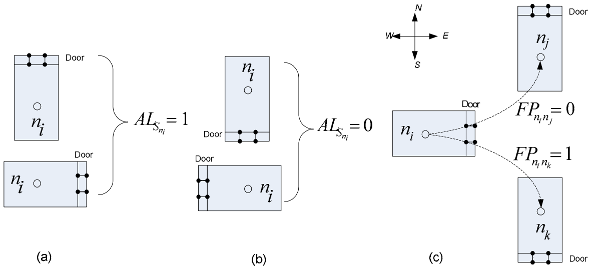

Fig. 1: Container orientations (a,b) and flip movements (c).

Since the orientation of containers is fixed in a single direction for both ships and trucks, path planning must consider the orientation at the initial and destination nodes. To guarantee the orientation at the destination node we model the alignment and changes in alignment during transportation (flip movements) as a container is transported from its initial node to destination node. Fig. 1 depicts the door alignment and the situations of changing direction via flip movements.

$AL_{S_{n_i}}$: $AL_{S_{n_i}} \in \{0,1\}$ is the box alignment of a container at node n_i. If a container door faces north (N) or east (E), then the container alignment is *aligned* and $AL_{S_{n_i}} = 1$. Else if, a container door faces south (S) or west (W), then the container alignment is *opposite,* and $AL_{S_{n_i}} = 0$.

$FP_{n_i n_j}$: $FP_{n_i n_j} \in \{0,1\}$ is a flip flag of each physical link regarding changing container's alignment during traversal of the link. Let $FP_{n_i n_j} = 1$ if container alignment is changed from node (n_i) to node (n_j), otherwise $FP_{n_i n_j} = 0$. The flip flag of each link is given by the map.

$AL^{REQ}_{J_k}$: required box alignment at destination node (n_D) for job $J_k : (J_k \in J)$. $AL^{REQ}_{J_k} \in \{Aligned, Opposite\}$ is dependant on the actual job type.

$AL^{Start}_{J_k}$: $AL^{Start}_{J_k} \in \{Aligned, Opposite\}$ is the initial alignment associated with job (J_k).

Let the timings for arrival and departure at each node in the trajectory be given by the ordered sets $T_{arrive} = \{t^1_{arrive}, ..., t^n_{arrive}\}$ and $T_{depart} = \{t^1_{depart}, ..., t^n_{depart}\}$ respectively, where $A \subset (L \cup N)$ is the set of all positions of a SC's trajectory, and the following constraints must be met:

1. If the trajectory's the first position $a_1 \in N$ is a **node** then, t^k_{arrive} is the arrival time and t^k_{depart} is the departure time at node $a_{2k-1} \in A$, where $k = (1,2,3,...,n)$;
2. If the first position $a_1 \in L$ is a **link** then, t^k_{arrive} is the arrival time and t^k_{depart} is the departure time at node $a_{2k} \in A$, where $k = (1,2,3,...,n)$;
3. $\forall t^k_{arrive} \in T_{arrive}, \forall t^k_{depart} \in T_{depart} : t^k_{depart} \geq t^k_{arrive}$.

The node, link, arrival and departure time ordered sets can be combined into the following representation:

$$\delta^{V_i} = \left[A, T_{arrive}, T_{depart}\right]^{V_i} = \left[\{a_1, ..., a_m\}, \{t^1_{arrive}, ..., t^n_{arrive}\}, \{t^1_{depart}, ..., t^n_{depart}\}\right]^{V_i}$$

$$where\ 2n - 1 \leq m \leq 2n + 1$$

This equation represents a planned trajectory of a SC(V) and associated timings for arrival (t^k_{arrive}) and departure (t^k_{depart}) at each node (a_k) in the path.

In order to achieve optimal task allocation of SCs at anytime, we must be able to accurately determine the relative timings for all jobs. Fig. 2 illustrates the relationship between SC position (relative to nodes) and the associated timing information for a job. Furthermore, as SCs move from the current position to the pickup node as part of an allocated job, it can be preempted and assigned to a different job. However, once it arrives at the pickup it cannot be assigned to a different job until it completes the current job.

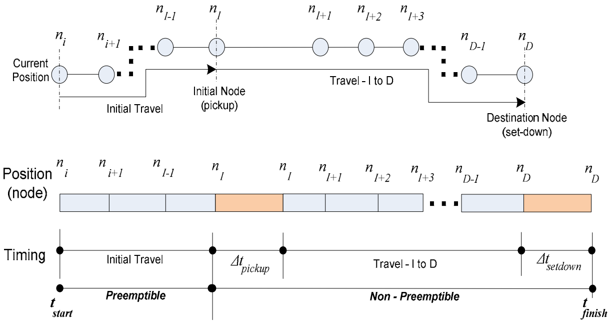

Fig. 2: Typical event timings for a SC performing a job.

D. *Objective Functions and Constraints*

The fundamental idea of implementing cost functions as part of the model formulation is to permit input of a desired work throughput, which can be translated by an additional controller into required rates and resourcing levels.

In this paper, we assume that the sequence of both import and export containers for ships and trucks are specified *a priori* and provided to the task allocation and path planning algorithms. The cost function for SC usage $V_i \in V$ is given by:

$$f_{V_i} = \left(t_{PF} - t_{PS} - t_{idle}\right)^{V_i} \tag{1}$$

Where, t_{PF} = Planned Finish (setdown) time of ***last*** job of the SC (V_i), t_{PS} = Planned Start time of *first* job (including initial travel time), t_{idle} = Total planned idle time.

Given the highly dynamic nature of the yard operations and the other seaport resources, the probability of unexpected events which result in delays to the short term traffic plan (schedule) can be quantified. As such, the robustness of a given traffic plan may be assessed by the level of *impact* such stochastic delay events have on the short term traffic plan. In order to improve the robustness of traffic plans, the following *bonus* function aims to encourage early start and finish times for both short jobs and long jobs. The bonus function is given by:

$$f_B^{J_k} = \int_{t_{start}^{J_k}}^{t_{finish}^{J_k}} \frac{1}{t - t_0} \times PT_{PUtoSD}^{J_k}(t)\,dt \tag{2}$$

Where, from Eq(2) and the problem definitions:

$$PT_{PUtoSD}^{J_k}(t) = \begin{cases} \mu^{J_k}, & t_{start}^{J_k} \le t \le t_{finish}^{J_k} \\ 0, & t_{finish}^{J_k} < t \; or \; t < t_{start}^{J_k} \end{cases}, \mu^{J_k} = \frac{MinTime^{J_k}}{t_{finish}^{J_k} - t_{start}^{J_k}} \tag{3}$$

Therefore;

$$f_B^{J_k} = \int_{t_{start}^{J_k}}^{t_{finish}^{J_k}} \frac{1}{t - t_0} \times \mu^{J_k}\,dt = \frac{MinTime^{J_k}}{t_{finish}^{J_k} - t_{start}^{J_k}} \times \ln \frac{t_{finish}^{J_k} - t_0}{t_{start}^{J_k} - t_0} \tag{4}$$

The bonus function expressed in Eq(4) aims to encourage early starting and early finishing of jobs, regardless of their duration. That is, jobs with short duration are treated equally as jobs with long durations. Combining Eq(1) and Eq(4) the overall objective is to minimise:

$$F = \lambda_1 * \sum_{V_i} f_{V_i} - \lambda_2 * \sum_{J_k} f_B^{J_k} \tag{5}$$

Where, λ_1 and λ_2 are the parameters to normalise the vehicle usage and *bonus* into dollars ($).

The constraints are:

$$\forall V_i \in V, \forall J_k \in J : x_{V_i J_k} = \{0,1\} \tag{6}$$

$$\forall J_k \in J : \sum_{V_i \in V} x_{V_i J_k} = 1 \tag{7}$$

$$\forall J_k \in J : t_{start}^{J_k} < t_{finish}^{J_k} \tag{8}$$

$$\forall \left(J_a, J_b\right) \in J : \alpha_{ab} = 1 \rightarrow t_{start}^{J_a} < t_{start}^{J_b} \tag{9}$$

$$\forall \left(J_a, J_b\right) \in J : \beta_{ab} = 1 \rightarrow t_{finish}^{J_a} < t_{finish}^{J_b} \tag{10}$$

$$\begin{aligned} &\forall V_i \in V \wedge \forall J_k \in J : x_{V_i J_k} = 1 \\ &\rightarrow \left(P^{V_i}\left(t_{start}^{J_k}\right) = n_I^{J_k} \wedge P^{V_i}\left(t_{finish}^{J_k}\right) = n_D^{J_k}\right) \end{aligned} \tag{11}$$

$$\forall t : P^{V_i}\left(t\right) \notin \bigcup_{j=1; i \ne j}^{|V|} S_{lock}\left(V_j, t\right) \tag{12}$$

$$\begin{aligned} &\forall J_k \in J, x_{V_i J_k} = 1 :\; AL_{J_k}^{Start} = AL_{J_k}^{Finish} \rightarrow \\ &\exists\{(a_x, t_{V_i\_arrive}^{a_x}, t_{V_i\_depart}^{a_x}), (a_y, t_{V_i\_arrive}^{a_y}, t_{V_i\_depart}^{a_y})\} \subset \delta^{V_i}, \\ &t_{start}^{J_k} \in (t_{V_i\_arrive}^{a_x}, t_{V_i\_depart}^{a_x}), t_{finish}^{J_k} \in (t_{V_i\_arrive}^{a_y}, t_{V_i\_depart}^{a_y}), \\ &x \le y - 1, (\sum_{i=x}^{i=y-1} \mathrm{FP}_{\mathrm{a_i} a_{i+1}}) MOD\ 2 = 0 \end{aligned} \tag{13}$$

$$\begin{aligned} &\forall J_k \in J, x_{V_i J_k} = 1 :\; AL_{J_k}^{Start} \ne AL_{J_k}^{Finish} \rightarrow \\ &\exists\{(a_x, t_{V_i\_arrive}^{a_x}, t_{V_i\_depart}^{a_x}), (a_y, t_{V_i\_arrive}^{a_y}, t_{V_i\_depart}^{a_y})\} \subset \delta^{V_i}, \\ &t_{start}^{J_k} \in (t_{V_i\_arrive}^{a_x}, t_{V_i\_depart}^{a_x}), t_{finish}^{J_k} \in (t_{V_i\_arrive}^{a_y}, t_{V_i\_depart}^{a_y}), \\ &x \le y - 1, (\sum_{i=x}^{i=y-1} \mathrm{FP}_{\mathrm{a_i} a_{i+1}}) MOD\ 2 = 1 \end{aligned} \tag{14}$$

$$\begin{aligned} &\forall t_{Start}^{J_k}, x_{V_i J_k} = 1 : \\ &\exists(t_{arrive}, t_{depart})^{P^{V_i}(t_{Start}^{J_k})}, t_{Start}^{J_k} \in (t_{arrive}, t_{depart})^{P^{V_i}(t_{Start}^{J_k})}, \\ &t_{depart}^{P^{V_i}(t_{Start}^{J_k})} - t_{arrive}^{P^{V_i}(t_{Start}^{J_k})} \ge \Delta t_{pickup} + \Delta t_{accelerate} + \Delta t_{decelerate} \end{aligned} \tag{15}$$

$$\begin{aligned}&\forall t_{Finish}^{J_k}, x_{V_iJ_k} = 1 : \\ &\exists(t_{arrive}, t_{depart})^{P^{V_i}(t_{Finish}^{J_k})}, t_{Finish}^{J_k} \in (t_{arrive}, t_{depart})^{P^{V_i}(t_{Finish}^{J_k})}, \\ &t_{depart}^{P^{V_i}(t_{Finish}^{J_k})} - t_{arrive}^{P^{V_i}(t_{Finish}^{J_k})} \geq \Delta t_{setdown} + \Delta t_{accelerate} + \Delta t_{decelerate}\end{aligned} \tag{16}$$

$$\begin{aligned}&\forall a_x \in \delta \cap L, \forall V_i \in V : \omega_{a_x a_{x+2}} = 1 \\ &\rightarrow t_{V_i\_depart}^{a_{x+1}} - t_{V_i\_arrive}^{a_{x+1}} \geq \Delta t_{decelerate} + \Delta t_{accelerate}\end{aligned} \tag{17}$$

$$\forall a_x \in \delta \cap L, \forall V_i \in V : t_{V_i\_arrive}^{a_{x+1}} - t_{V_i\_depart}^{a_{x-1}} \geq TT_{\mathrm{a}_{\mathrm{x-1}} a_{x+1}} \tag{18}$$

where, (6) and (7) ensure that each job is performed by one and only one SC; (8) ensures that the job start time must precede the corresponding job finish time; (9) guarantees a feasible picking sequence between jobs, such as stacked jobs; (10) guarantees a feasible set-down sequence between jobs. Boxes are not eligible for set-down unless they are next in the job sequence and there will be an empty destination node; (11) ensures that both start position and destination position are specified for SC jobs; (12) provides collision avoidance between SCs; (13) and (14) ensure containers' alignment changes are valid within trajectories; (15) and (16) ensure SCs perform a feasible action: 'pickup' or 'setdown' on a node in light of the motion profile; (17) ensures that SCs should spend some time on reversing or making a turn, i.e. there are acceleration and deceleration processes; (18) ensures that SCs meet minimum travel time (*TT*) when travelling from a node to another.

III. Solution and Simulation

This section describes a feasible solution strategy and results for the objective function. Our collision-free path planner implements a prioritised multi-vehicle path planning algorithm [9]. This planning algorithm is extended from a single-vehicle time-window-based algorithm [10] to calculate feasible time and cost windows for each vehicle to arrive at and depart from each node (subject to time-dependent node and link availability). Such time windows are propagated iteratively from the known starting time of the vehicle at the start node until a feasible arrival window is found at the destination. The key feature of this algorithm is that the paths generated will consider the motion of all other active SCs and as a result will cause them to go around or give way (via waiting at a node or shunting aside and subsequently resuming) to SCs with already planned paths. Significantly, this is more complex than simplified vehicle routing problems where path lengths only need to be calculated once, regardless of the changing occupancy of the various nodes and links in the graph.

Furthermore, in order to increase the accuracy of task scheduling and viability of collision-free paths, typical motion dynamics of the SCs are considered in our model. The velocity of every SC is zero at its initial position, during box pickup and box setdown operations and during a 3-point turning maneuver. Therefore, SCs undergo finite acceleration and deceleration during their motion, which we reflect in the constraints of Eq(15) - Eq(17). An example path for a SC travelling from the pickup node and then onto the setdown node is illustrated in Fig. 3.

Now, Fig. 4 shows the motion profile applied to the example path in Fig. 3. Here, we can see that during the first 7 nodes the velocity increases from 0ms^{-1} to the maximum velocity (8.3ms^{-1}). Then, deceleration occurs as the SC performs a three-point turn at node 16, where the velocity is 0ms^{-1}. The SC then accelerates to maximum velocity and travels to the setdown node where it begins deceleration at node 49 and stops at node 56 to perform the box setdown operation.

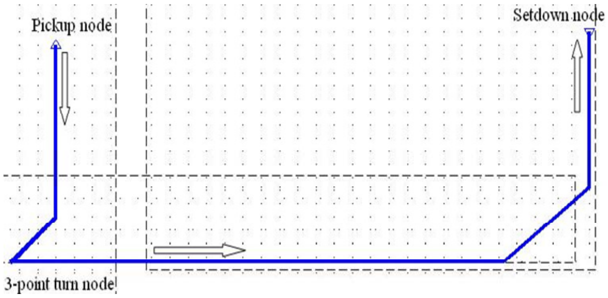

Fig. 3: A planned path from pickup node to setdown node consiting of 56 nodes. A 3-point turn occurs at node 16 (nodes are not uniformly spaced in the map).

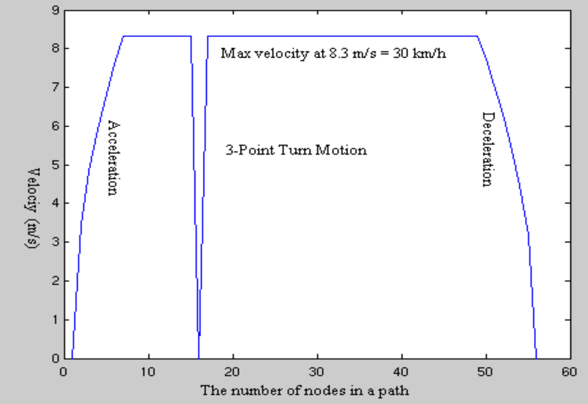

Fig. 4: Example of a motion profile, which is applied to a planning path for a SC (nodes are not uniformly spaced in the map).

To validate the efficacy of the model and verify the feasibility of solutions, we developed a simple algorithm to plan and schedule jobs sequentially using a greedy heuristic based on nearest-vehicle-first. However, this approach does not guarantee optimal job allocation, since the current path planner computes paths sequentially using the existing time-windows for each planned path. The approach (Fig. 5) was simulated using a fleet size of (4, 8, 12, 16, 20) SCs and a job-horizon of *maxJobHorizon* = 5.

Collision-free paths where successfully planned for the all jobs and SCs for the problem. If the total number of jobs in the initial job list exceeded *V* × *maxJobHorizon*, then a simple form of replanning can be performed for those remaining jobs. This computation occurs online and feasible jobs are then injected into the running job pool for execution by the assigned SC.

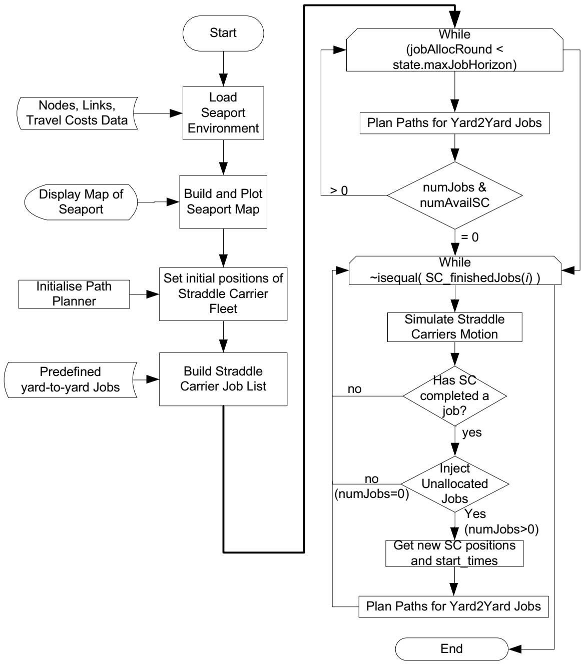
Start
Nodes, Links, Travel Costs Data
Load Seaport Environment
Display Map of Seaport
Build and Plot Seaport Map
Initialise Path Planner
Set initial positions of Straddle Carrier Fleet
Predefined yard-to-yard Jobs
Build Straddle Carrier Job List
While (jobAllocRound < state.maxJobHorizon)
Plan Paths for Yard2Yard Jobs
> 0
numJobs & numAvailSC
= 0
While ~isequal(SC_finishedJobs(i))
Simulate Straddle Carriers Motion
no
Has SC completed a job?
yes
no (numJobs=0)
Inject Unallocated Jobs
Yes (numJobs>0)
Get new SC positions and start_times
Plan Paths for Yard2Yard Jobs
End

Fig. 5: Flow diagramming for sequential job allocation and job injection approach based on a greedy nearest-vehicle-first heuristic.

Table 1 shows the experimental results of different combinations of SCs for 100 jobs. In these experiments, $\lambda_1 = \$5/1000$ time units and $\lambda_2 = 0.1 \times Bonus(\$)$ to normalise total vehicle usage and bonus function value into dollar units.

TABLE 1: COMPUTATION OF THE OBJECTIVE FUNCTION Eq.**(5)**

	SCUsageTotal	TotalBonus	OverallCost
4-Vehicle	53425	746	**52679**
8-Vehicle	56830	1049	**55781**
12-Vehicle	55280	1254	**54026**
16-Vehicle	57195	1426	**55769**
20-Vehicle	57800	1533	**56267**

The experimental results show an increasing value of TotalBonus being awarded with increasing fleet size. This is due to earlier start and finish times for allocated jobs, however the total SC usage cost also increases. The net effect, without any additional optimisation shows a slight increase in overall costs.

IV. Conclusion and Future Work

We have presented a mathematical model and simulation results of proposed solutions for the yard operations within an automated seaport container terminal. Several difficult challenges have been addressed, including the formulation of an objective function which minimises total yard-vehicle usage, while producing robust traffic plans by encouraging both early starting and finishing of jobs. Moreover, the formulated model incorporates many of the difficult practical challenges of an automated container terminal, including usage of a large fleet of autonomous vehicles, multiple levels of container stacking and sequencing, variable container orientations, vehicular dynamics and collision-free path planning.

Our future work will focus on two main areas. Firstly, we aim to extend the model and simulator to include a set of quay cranes and a set of trucks docked in the truck import area. Secondly, we aim to refine the existing mathematical model and simulation environment to a point where it is effectively shared with other interested researchers. This would permit application of various optimisation methods and computation of feasible and optimal online job schedules. Finally, we would be able to compute an online schedule to the replanning problem for the entire seaport terminal environment and its resources.

Acknowledgment

This work is supported by the ARC Linkage Grant (LP0882745), the ARC Centre of Excellence program (funded by the Australian Research Council (ARC) and the NSW State Government), the Patrick Stevedores Holdings and the University of Technology, Sydney, Australia.

References

[1] I. Vis and I. Harika, *Comparison of vehicle types at an automated container terminal*. New York: Springer, 2005.

[2] D. Steenken, S. Vob, and R. Stahlbock, *Container terminal operation and operations research – a classification and literature review*. New York: Springer, 2005.

[3] I. Vis and R. Koster, "Transshipment of containers at a container terminal: An overview," *European Journal of Operational Research,* vol. 147, pp. 1-16, 2002.

[4] A. Kozan and P. Preston, *Mathematical modelling of container transfers and storage locations at seaport terminals*. Berlin: Springer, 2007.

[5] S. Hartmann, *A general framework for scheduling equipment and manpower at container terminals*. New York: Springer, 2004.

[6] G. Nelmes, "Container Port Automation," in *The 5th International Conference on Field and Service Robotics*. vol. 1, P. Corke and S. Sukkarieh, Eds. Port Douglas, Australia: Springer, 2005, pp. 3 - 8.

[7] S. Yuan, H. Lau, D. K. Liu, S. D. Huang, G. Dissanayake, D. Pagac, and T. Pratley, "Simultaneous dynamic scheduling and collision-free path planning for multiple autonomous vehicles," in *Proceedings of the 2009 IEEE International Conference on Information and Automation*, Macau, China, 2009, pp. 522-527.

[8] D. K. Liu and A. K. Kulatunga, "Simultaneous planning and scheduling for multi-autonomous vehicles," in *Evolutionary Scheduling*. vol. 49, K. P. Dahal, K. C. Tan, and P. I. Cowling, Eds. Berlin: Springer-Verlag, 2007, pp. 437-464.

[9] H. Lau, T. Pratley, D. K. Liu, S. Huang, and D. Pagac, "An implementation of prioritized path planning for a large fleet of autonomous straddle carriers," in *18th Triennial Conference of the International Federation of Operational Research Societies (IFORS),* Sandton, South Africa, 2008.

[10] J. Halpern, "Shortest Route with Time Dependent Length of Edges and Limited Delay Possibilities in Nodes," *Mathematical Methods of Operations Research,* vol. 21, pp. 117-124, 1977.