Adaptive Cost Coefficient Identification for Planning Optimal Operation in Mobile Robot based Internal Transportation

Pragna Das, Lluís Ribas-Xirgo, Member, IEEE

Abstract—Decisions in mobile robot based logistic systems can be improved based on knowledge of real-time state of individual parts and environmental factors. In case of battery operated mobile robots, the cost of performance depends not only on physical state but also on state of charge of batteries. The knowledge about these factors can be obtained through cost coefficients by individual robots. Our work focuses on identifying these cost coefficients in a mobile robot used in internal transportation, which can be fed to any standard planning algorithm (like Dijkstra) to optimize total cost of operation. Travel time is one such type of cost coefficients. With suitable predictions of these travel times the cost involved to traverse from one node to another can be known. Suitable state-space model is formulated and Kalman filtering is used to estimate these travel time. Experimental validation of the efficacy of these travel times has been conducted by using them as weights for edges in standard route planning algorithm. Results show that when travel times is used as weights to compute path, the total traversing cost of paths has been reduced by 15% on average in comparison to that of paths obtained by heuristics costs.

Index Terms—Mobile robot, autonomous systems, planning and co-ordination, cost parameter, parameter estimation, cost efficiency, Kalman filtering

I. INTRODUCTION

MOBILE ROBOT (MR) based systems used for internal logistics in factories demand cost efficient decisions on planning and co-ordination. Decisions on planning becomes better when they are based on information about the MR's batteries and environmental factors, obtained during operation of the system [3], [8], [14]. This idea is explained in the following example. Figure 1 illustrates a scaled down automated internal transportation system, typically used for logistics in factories and executed by MRs. In this example, all MRs can execute only one task at a time. Let, at $t_i$, the path computed for $A_1$ to carry some material to $P_1$ is marked by the dotted line. Again, at time $t_j$ ($j > i$), $A_1$ needs to carry same material to $P_1$. But now, the battery capability of $A_1$ has decreased due to execution of previous tasks and the condition of the given path has deteriorated (marked by dotted rectangle). Hence, equal amount of time and energy as previous would not be sufficient to reach $P_1$ at $t_j$ by $A_1$. At this juncture, decision on routes can be improved considering utility [14] like capability, or fitness et cetera of completing the task. This capability can be directly measured by the time(s) needed to perform task(s) by individual robots [17] in the system. In the above example, time required to reach one node from another node can be measured to denote the capability or fitness of the MR. Hence, these quantities are formalized as cost coefficient in order to incorporate the different conditions of the environmental, physical and mechanical parts of any MR into the control decision steps. In Figure 1 the real cost, considering the floor condition, required to traverse from one node to another can lead to a better optimal path to reach $P_1$. This cost can be known directly by estimating the time to go from one node to another, i.e.-time to travel between two nodes through the connecting edges. The travel time is calculated considering the difference between the departure from one node and reaching the next node and thus travel time is not dependent on the shape of the edge, rather it depends on the time taken to traverse the edge. These travel times which includes the dynamically changing factors, when considered as cost of traversing the edges, can produce a different path than previous, for example the path marked by solid line can be a better decision to traverse to $P_1$.

Aforementioned cost coefficients like travel time of edges can be obtained irrespective of the kind of task the robot needs to perform. In case of autonomous logistics, travel time between nodes can be thought of one such cost factor which determine the utilities based not only on internal factors of each robot but also on environmental factors, while the the task is traversing from one port to another. On one hand, they arise locally at each MR due to action of actuators,
wheels and other mechanical factors, but on the other they are
significantly influenced by environmental factors like battery
capacity (in case of battery powered MRs), conditions of the
floor, conditions of material to be transported, performance
and behavior of other AGVs, et cetera, as all or most of these
factors determine the state of the robot at every instance of
time. Influence of factors like friction forces of floor, slope,
mechanical part can be corrected by local control on individual
MR (lower levels), but factors like traffic condition, conditions
of material, behavior of other MRs are beyond the scope of
control by lower levels. Hence, considering cost coefficients
at lower levels of actuation and control cannot make better
control decisions. So these parameters are investigated at
higher level to utilize them efficiently. These costs are time
dependant and have sources of error from battery exhaustion,
surface condition of shop floor, wear and tear of tyre, et cetera.
Hence over the passage of time, the values will change. For
example, Part (b) in Figure ?? plots the progressive mean of
observed values of travel time for an mth edge first only with
the change of state of charge of batteries and Part (c) with
both the change of state of the charge of batteries and the
floor condition from rough at the beginning to smooth. Part
(a) of Figure ?? plots the cell voltage over time of Li-ion
batteries.

The plot of progressive mean of travel time of (b) shows
that the values increase first, then steadily decrease and then
increase gradually till complete discharge. But the longer
increase of values of travel time in (c) can be attributed to
the rough floor, because at equal battery capacity in both
cases, more energy is required to traverse in rough surface.
Plot of the same arc travel time in different conditions of floor
demonstrate that travel time can reflect not only state of charge
of batteries [9] but also environmental conditions.

The efficacy of the travel times in planning is evaluated
by estimating them for each required edge while making
decisions to compute routes. The travel time is considered
as weight of connecting edges to facilitate standard planning
algorithm like Dijkstra to compute least cost consuming path.
Thus, Dijkstra’s algorithm is modified by using travel time
as cost of edge, instead of heuristic cost based on distance
for edges. The total travelling costs of two categories of
paths obtained by heuristically gathered cost and real travel
time, irrespective of the route planning method, are calculated.
Average total costs of path, obtained using static estimation
of travel time ( variation of edge travel cost over time is
not considered) is roughly 5% less that of paths, obtained
by heuristics costs. (Section [V]). However, travel times of
edges (edge costs) vary along time and require to be predicted
accordingly during path planning. A good estimation method
to accurately predict travel times requires their histories, which
can be collected progressively during MR operation. The
estimation process start with mean of data obtained from
legacy and real observations are obtained during the operation.
Thus, the filtering method cannot generate the best estimates
at initial few iterations. The estimates get improved over
time. Real travel times are obtained by these estimations
which can produced different paths than that of other costs
like theoretical, heuristic and experimental. In fact, estimating
traversal time by Kalman filtering shows that heuristic edge
costs can underestimate total costs and, thus, can lead to non-
minimal paths.

The contribution of this paper is twofold. Firstly, travel
time of edges is identified as a suitable cost coefficient
considering an analogy to real, automated and fully functional
plant. Secondly, these identified travel times are estimated both
statically and dynamically. Further to this, they are utilised in
a planning decisions to culminate into cost efficient optimal
results for an MR.

The next section highlights the background and previous
state of the art (Section [II]). Section [III] formulates the
problem in the light of an internal logistic system with path traversal
as a task. Section [IV] explains the prototype platform and
other details for the system used to conduct experiments. Two
experiments and their results are elaborated in Sections [V] and
[VI] with Algorithm [I] elaborating on the proposed approach
which incorporates modification over Dijkstra’s algorithm.
Section [VII] concludes with discussions and future directions
of investigation.

II. BACKGROUND

So far, path planning has been solved in two distinctly dif-
terent approaches for autonomous robots. Sampling methods
perform reasonably well in solving intricate path planning
problems in static and dynamic environment for a single
robot [12]. The vehicular dynamics are considered as state
in these approaches and the minimum cost path is obtained
by spanning the search tree based on the distance between
the current state and goal state. Although Suh and Oh in [23] and Achtelik et. al in [1] have used Gaussian process as the cost of the path to incorporate environmental parameters, the search mandates to conceive the vehicular dynamics of the robot. Also, sampling based methods are blocked into local minima and uncertainties in the environment hinder successful results [12]. Further, heuristic approaches like Artificial Neural Network (ANN) [10], Genetic Algorithm (GA) [2], Particle Swarm optimisation (PSO) [24], Ant Colony Optimisation (ACO) [6], et cetera can adapt to uncertainties and changing environment. But, they are computationally expensive which is a major concern for robotic control units equipped with limited resources and real-time constraints [15]. The vehicular mechanical factors and environmental factors are incorporated in travel times in this work to mitigate the identification of the vehicular dynamics. The path is computed in higher level of robotic control and paths are broken down to simple vehicular commands for movements and communicated to the lower-levels of control. Also, simple, deterministic and computationally inexpensive Dijkstra’s algorithm is deployed and travel times are incorporated with it to decide paths of minimum cost.

On the other hand, motion planning and task planning, serves as specific coordination problems in MRS. Cost coefficients are usually computed heuristically before hand (offline) using Markov Decision Processes (MDPs). However, motion planning and task planning, as a specific case of coordination problem, are typically NP-hard and are addressed to find tractable and good solutions [11] and are mostly treated as individual specific problems in field of MRS [5], [13], [22].

In this work, path planning has been considered as an example of planning, where travel times are utilized as costs to obtain optimal path in a single robot.

III. PROBLEM FORMULATION

In this work, the focus is on one MR. Also, traversing a path is considered as a task (Section I).

A path \( P \) for a robot is usually defined as,

\[
P = \{(n_a, n_b), (n_b, n_c), (n_c, n_d), (n_d, n_e), \ldots\ldots\}\]

(1)

where \( n_p \) is any node. \( P \) can be also expressed in terms of connecting edges as

\[
P = \{(a_{a,b}), (a_{b,c}), (a_{c,d}), (a_{d,e}), \ldots\ldots\}\]

(2)

where \( a_{p,q} \) is any edge.

As explained in Section I, each edge in the floor map is associated with some cost in terms of energy exhaustion and others. Hence, traversing a defined path incurs several travel costs for all edges in the path. Thus time to traverse an edge by a MR can be conceptualized as its cost coefficients. Let \( X_{p,q}(e, f) \) denote edge cost from \( n_p \) to \( n_q \), where \( e \) denotes dependency for state of charge of batteries and \( f \) denotes dependency for frictional force of the floor. Now, the total cost of traversing \( P \) can be written in a form \( P_e \) as,

\[
P_e = \{(X_{a,b}(e, f), X_{b,c}(e, f), X_{c,d}(e, f), \ldots\ldots)\}

(3)

In equation 3 it is shown that a total path cost is dependent on all edge costs and each edge cost is denoted by its travel time.

Thus, the \( X \) denotes general travel time of any edge. Also, travel time of any edge \( X \) depends on all the previous edges the robot has already traversed. The reason being the discharge of batteries and (or not) possible change of environment. Thus, travel time \( X \) becomes a function of \( k \) where, \( k \) = number of time a MR has performed the task of traversing any edge.

Hence, \( X(k) \) is estimated with respect to increase in \( k \) for any edge and used as weight of edge to compute path. The estimated value of \( X(k+1) \) depends only on \( X(k) \) and the observation of \( X \) at \((k+1)\). These experiments and results are explained in Section VI. Observations of all possible \( X(k) \) for all possible \( k \) needs to be made for this above estimation for a single MR.

However, this is not only cumbersome but also impractical to gather such huge amount of observation. This estimation is static as \( X \) is estimated without considering its variation with the total elapse of time from start of system. The static estimation approach is progressed to a different model. Observations of all possible \( X(k) \) for all possible \( k \) is not needed in the latter. A window of previous values of \( X \) is decided to form a state vector. This state vector is estimated on every \( k \) to find the estimated value of \( X(k+1) \). Thus, the current value of \( X \) is estimated depending on the previous \( X \)’s i.e.-travel times of edges which are already being found to form the path, along with the variation of exploration of \( X \) due to elapse of time. Thus, \( X \) values are dynamically estimated considering its variation over elapse of time. Moreover, the model is allowed to gather the possible values of \( X \) itself from the beginning of first call of path planning and use these values to estimate current value. This experiment is elaborated in Section VI.

IV. PROTOTYPED INTERNAL TRANSPORTATION SYSTEM

A prototype scaled down internal transportation system is developed with all essential constituting parts like MRs, tasks, controller architecture and the environment adhering to minute details. The floor is described by means of a topology map \( \mathcal{G} = \{\mathcal{V}, \mathcal{E}\} \), where each port and bifurcation point corresponds to some node \( n_r \in \mathcal{V} \) and each link between two nodes forms an edge \( a_{r,s} \in \mathcal{E} \). Part (a) of Figure 3 depicts a portion of the whole prototype, where, notation like \( n_{26} \) designates a node and \( a_{26,27} \) a edge. Topology maps are generated taking reference from the grid map generated by results of Self Localisation and Mapping (SLAM) in [4] based on a simple assumption, that each free cell in the grid map corresponds to a node in the graph. A selected representative portion from each of the three sections of of Coca-Cola Iberian Partners in

![Scale down prototype platform](image)

---

Thus, the \( X \) denotes general travel time of any edge. Also, travel time of any edge \( X \) depends on all the previous edges the robot has already traversed. The reason being the discharge of batteries and (or not) possible change of environment. Thus, travel time \( X \) becomes a function of \( k \) where, \( k \) = number of time a MR has performed the task of traversing any edge.

Hence, \( X(k) \) is estimated with respect to increase in \( k \) for any edge and used as weight of edge to compute path. The estimated value of \( X(k+1) \) depends only on \( X(k) \) and the observation of \( X \) at \((k+1)\). These experiments and results are explained in Section VI. Observations of all possible \( X(k) \) for all possible \( k \) needs to be made for this above estimation for a single MR.

However, this is not only cumbersome but also impractical to gather such huge amount of observation. This estimation is static as \( X \) is estimated without considering its variation with the total elapse of time from start of system. The static estimation approach is progressed to a different model. Observations of all possible \( X(k) \) for all possible \( k \) is not needed in the latter. A window of previous values of \( X \) is decided to form a state vector. This state vector is estimated on every \( k \) to find the estimated value of \( X(k+1) \). Thus, the current value of \( X \) is estimated depending on the previous \( X \)’s i.e.-travel times of edges which are already being found to form the path, along with the variation of exploration of \( X \) due to elapse of time. Thus, \( X \) values are dynamically estimated considering its variation over elapse of time. Moreover, the model is allowed to gather the possible values of \( X \) itself from the beginning of first call of path planning and use these values to estimate current value. This experiment is elaborated in Section VI.

IV. PROTOTYPED INTERNAL TRANSPORTATION SYSTEM

A prototype scaled down internal transportation system is developed with all essential constituting parts like MRs, tasks, controller architecture and the environment adhering to minute details. The floor is described by means of a topology map \( \mathcal{G} = \{\mathcal{V}, \mathcal{E}\} \), where each port and bifurcation point corresponds to some node \( n_r \in \mathcal{V} \) and each link between two nodes forms an edge \( a_{r,s} \in \mathcal{E} \). Part (a) of Figure 3 depicts a portion of the whole prototype, where, notation like \( n_{26} \) designates a node and \( a_{26,27} \) a edge. Topology maps are generated taking reference from the grid map generated by results of Self Localisation and Mapping (SLAM) in [4] based on a simple assumption, that each free cell in the grid map corresponds to a node in the graph. A selected representative portion from each of the three sections of of Coca-Cola Iberian Partners in
Bilbao, Spain are extracted to form three topological maps. They are provided in Figure 4 Part (a) of Figure 4 illustrates Map 1 which is a representative of winding racks in the warehouse facility, Part (b) shows Map 2 which represents randomly placed racks and Part (c) shows Map 3 which represents racks organized in a hub. The scaled robots are built controller, ultrasound sensor and a camera, as illustrated in Part (b) of Figure 3. The DC servo motors drive the wheels of the MR and Li-ion batteries energize them. Each MR has its individual controller in decentralized architecture [19], [20]. Travel times for three different length of arcs in all three maps and four different conditions of surface are recorded till complete exhaustion of batteries. This generates observation data for all possible $X$ for all $k$s. This cumbersome process of acquiring the observations for generating on-line estimates is mitigated by a non-linear functional model of $X_{p,q}$ incorporating its evolution over passage of time and also allowing to gather information about $X_{p,q}$ for different arcs in the map gradually with time.

V. EXPERIMENT I: USING STATIC ESTIMATES OF TRAVEL TIMES IN ROUTE PLANNING

A. Procedure

The state-space model provided in equations 4 and 5 is used to estimate travel times, considering them as cost of edges. From now $X_{p,q}$ will be written as $X$ for simplicity.

$$X(k) = X(k-1) + \omega(k)$$ (4)

$$Y(k) = X(k) + \eta(k)$$ (5)

The state vector in equation 4 is a single variable $X$ depending on $k$, $k$ being the number of edges already found in the path (Section III). Hence, $X$ is estimated over and over again for different connecting edges of every new exploring node. $Y(k)$ in equation 5 is the observation variable for $X$. This model involves two error terms $\omega(k)$ and $\eta(k)$ which are independent and normally distributed. According to equations 4 and 5 $X(k)$ depends only on the travel time of edge between current node and its predecessor i.e.-$X(k-1)$, though in reality, it depends on $X$s for all the previous edges in the path and its own variation over the time. Thus, this process of estimation is static. (Section III). Equations 6 and 7 are obtained after applying Kalman Filtering method [21] on equations 4 and 5.

$$\hat{X}^{-} = \hat{X}(k-1)$$

$$P^{-} = P(k-1) + \sigma_{\omega}^{2}$$ (6)

$$K(k) = P^{-}(k)/[P^{-}(k) + \sigma_{\eta}^{2}]$$

$$P(k) = P^{-}(k) - [P^{-}(k)^{2}]/[P^{-}(k) + \sigma_{\eta}^{2}]$$ (7)

$$\hat{X}(k) = \hat{X}^{-}(k) + \omega(k)$$

where

$$\omega(k) = Y(k) - \hat{X}^{-}(k)$$

$\hat{X}^{-}(k)$ produces the apriori value of $X$ and $P^{-}$ produces the associated covariance, $\sigma_{\omega}^{2}$ being the covariance of process noise $\omega(k)$. $\hat{X}(k)$ provides the predicted estimate of $X(k)$, as $\hat{X}^{-}(k)$ is corrected in equation 7 with the help of Kalman Gain $K(k)$. $P^{-}(k)$ provides the associated covariance matrix, $\sigma_{\eta}^{2}$ being the covariance of the observation noise $\eta(k)$. For example, a sample route computation is illustrated in Figure 5. Let $n_{a}$ be source and $n_{w}$ destination at $P16$. So, path computation using Dijkstra’s algorithm starts at $n_{a}$ with its neighbors $n_{b}, n_{c}$ and $n_{d}$. So, $X_{a,b}$, $X_{a,c}$, $X_{a,d}$ are required to be estimated at $k$, when $k$ is 1 as this will be first edge being traversed.

$$\hat{X}(0) = E[X(0)]$$ (8)

$$P(0) = E[(X(0) - E[X(0)])(X(0) - E[X(0)]^{T})]$$ (9)

We use equation 6 to obtain $\hat{X}^{-}(1)$ for $X_{a,b}$, $X_{a,c}$, $X_{a,d}$ separately depending on $X(0)$ using equation 8. Similarly, we get separate $P^{-}(1)$ using equation 9. Next, we obtain $\hat{X}(1)$ (estimate) and $P(1)$ for $X_{a,b}$, $X_{a,c}$, $X_{a,d}$ using equation 7. Comparison of estimated values of $X_{a,b}$, $X_{a,c}$, $X_{a,d}$ will provide the least cost of traversing from $n_{a}$ to any of its neighbors. Let, the least cost edge be $a_{a,c}$. So $n_{a}$ will become the predecessor of $n_{c}$, i.e.-to reach $n_{c}$, the edge should come from $n_{a}$. When $n_{c}$ will be explored, the value for $k$ is 2 as $n_{c}$ has 1 predecessor. The next least cost edge from $n_{c}$ in the path is required to be known. Thus, $X_{c,e}$, $X_{c,f}$, $X_{c,g}$ needs to be estimated. Thus, observation $Y(k)$ of $X$ at current $k$ is required to estimate $X$. Thus observation values for travel costs of all possible $X$s for all possible $k$s were collected. This above process is explained in Algorithm 1. This static experiment is conducted to verify that weights of edges can be estimated online during exploration of Dijkstra’s algorithm using a state-space model. Also, it is verified that the estimated values of $X$ are correct and real through this experiment, as the values can be compared to real observations.

B. Results

Paths are computed repeatedly for 20, 40, 60 and 80 times in each topological graph (Figure 4) using both original Dijkstra’s algorithm using Euclidean distance based heuristic weights of edges and the modified one (Algorithm 1). The choices of source and destination are fed from the decided list of sources and destinations for each call of route computation. Total path costs obtained using heuristic weights are compared with paths obtained using Algorithm 1. The vertical bars of Eucl and SEC in Figure 6 represent the average total path costs for heuristic cost based routes and static estimates based routes respectively. Vertical bar Eucl shows that average total path costs never change with increase in number of repetitive calls, as heuristic weights do not change over time and does not reflect the true cost of traversal.

Vertical bar SEC shows that average total path costs obtained by Algorithm 1 is 5% less in case of Map 2 and Map 3 and 2% less in Map 1 than that of heuristic cost based Dijkstra’s algorithm. Average total path costs increases with number of repetitions as shown by vertical bar SEC, as duration of performance increases with increase of repetitions. This happens due to the dependency of current edge cost on previous edge cost (equations 4 and 5). But, this variation does not truly reflect the variation of travel time due to time-varying factors.
VI. EXPERIMENT II: USING DYNAMIC ESTIMATES OF TRAVEL TIMES IN ROUTE PLANNING

A. Procedure

The bi-linear model [18], provided in equation (10), is used to model the change of travel costs depending upon all the previous travel costs. $X$ is formed as a function of its past histories over $k$, considering the progressive change $\xi$ with respect to $k$. After start of computing a path, the real travel time of edges are recorded when the MR actually traverses it. This travel times of edges are used as the observation values for the next call of path planning. Thus observation values of travel times of each edge is grown during run-time.

$$s(k) = F(s(k-1))s(k-1) + V\xi(k) + G\omega(k-1)$$  \hspace{1cm} (10)

$$Y(k) = Hs(k-1) + \xi(k) + \eta(k)$$  \hspace{1cm} (11)

The state transition matrix $F$ in the equation (10) has the form of

$$F = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 1 \end{bmatrix}$$

The number of rows of $F$ is given by $(2^\text{regression_no} + 1)$. The $\psi$ terms in $F$ are denoted as in equation (12)

$$\psi_l = b_l + \sum_{i=1}^{l} c_{li}X(k-i)$$  \hspace{1cm} (12)

All the $\phi$ terms in $F$ are constants. The term $\mu$ is the average value of $X$ till $k$. Also, the matrix $V$ in (10) is denoted as

$$V = \begin{bmatrix} 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

The number of rows of $V$ is again given by $(2^\text{regression_no} + 1)$. The matrix $H$ in (11) is denoted as

$$H = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Kalman filtering is applied on the state-space model (equa-
approach is different from Algorithm 1 in the way the
of $x_n = (1 - n)$ travel cost has been known connecting
Exploration proceeds with average travel cost for the edges.
be formed as minimum 2 previous travel costs are needed.
In Figure 5, the path computation starts at $x_n$ and so
all edges arising out of $x_n$ to obtain $X$ for each edge. From
Equations 16 and 17, $s(0)$ and $P(0)$ can be obtained. Let at $x_n,$
4. Hence, $X(3)$ will be travel cost from $x_n$ (predecessor
of $y_n$) to $x_n,$ $X(2)$ will be travel cost from $x_n$ (predecessor
of $n_e$) to $n_e,$ $X(1)$ will be travel cost from $x_n$ (predecessor
of $n_e$) to $n_e.$ Thus, $s(3) = (1,$ $\xi(2),$ $\xi(3),$ $X(2),$ $X(3))^T$ and $s(4) = (1,$ $\xi(3),$ $\xi(4),$ $X(3),$ $X(4))^T$ needs to be computed. This
approach is different from Algorithm 1 in the way the $X$ is
estimated.

B. Results
The process of path computation is exactly similar to previous
experiment. Only difference is in the estimation procedure of
$X,$ which is based on the bilinear state space model.
The $b$ and $c$ are chosen as normal distribution. Along with
the repetitions of path computations, the value $\phi,$ mean and
covariance of $b$ and $c$ are increased from -0.4 to 0.4 and from
-0.2 to 0.2 respectively. Negative values of $\phi$ produced too
high estimates while values greater than 0.2 produced negative
estimates. Similarly, mean and covariance values less than 0.1
produce high estimates and more than 0.1 produce negative
estimates. Thus, 0.2 is found as the suitable value of $\phi$ and
$N(0,1,0.1)$ suits for both $b$ and $c.$ Also, the $regression_{no}$
from 2 to 9 for each of 20, 40, 60 and 80 repetitive com-
putation and average total path costs obtained on each case
are plotted in Figure 6. The vertical bars marked from $Reg_2$
to $Reg_9$ represent the average total path costs for dynamic
estimates based routes, which shows that they are 15% less
on average than heuristic euclidean cost for all three maps
in each set of repetitions. This difference is increased with
the increase of $regression_{no},$ though the rate of increase
is low, as the change of $X$ itself is not broadly spread with
standard deviation of 0.219 on average. The average total path
cost increases with increase in number of repetitions as edge
tavel cost increases with elapse of time. The observation $Y(k)$
developed during run-time is considered as signal and the
values of $\omega$ are modified to increased the Signal-to-Noise Ratio
(SNR) from 10dB to 50 dB along with the repetitions of path
planning. The vertical bars marked 10dB, 25dB and 50dB in
Figure 6 plots the average path costs obtained by changing the
SNR for each $regression_{no},$ which shows that with the
increase of SNR, the average travel cost decreases.

C. Path comparison
Part (a) of Figure 6 plots 3 single paths PathA, PathB and
PathC obtained from Dijsktra’s algorithm based on heuristic
Algorithm 1 Using static estimation of travel time in Dijkstra’s algorithm

1: function INITIALISE_SINGLE_SOURCE(\(V, s\)) \(\triangleright\) Where \(V\) - list of nodes, \(s\) - source, returns \(d[v]\) - attribute for each node, \(\pi[v]\) - predecessor for each node
2: for each \(x_i \in V\) do
3: \(\pi[x_i] = infinity\)
4: \(d[x_i] = NIL\)
5: end for
6: end function

7: function FIND_PREV((u, s)) \(\triangleright\) input: \(u\)-current node, \(s\)-source node, returns: \(prev\)/-predecessor of \(u\), \(noPred\) - number of predecessors till \(s\)
8: \(prev = compute\) predecessor of \(u\)
9: \(noPred = count\) of predecessors till \(s\)
10: end function

11: function KF((pW, k, Y(k))) \(\triangleright\) input: \(pW\)-value of travel time at \(k\)-1, \(k\)-instance for estimation, \(Y\)-observation variable, Returns: \(X(k)\)-travel cost from \(u\) to \(v\)
12: Apply KF on state-space model to obtain \(\hat{X}(k)\)
13: end function

14: function FIND_COST(u, v, k, pW, Y(k)) \(\triangleright\) Input: \(u\)-current node, \(v\)-neighbor node, \(k\)-instance of estimation, \(pW\)-cost between \(prev\) and \(u\), \(Y(k)\)-observation of travel time between \(u\) and \(v\) \(\triangleright\) Returns: \(w\)-estimated travel time (cost) from \(u\) to \(v\)
15: \(w = KF(pW, k, Y(k))\)
16: end function

17: function RELAX(u, v, w) \(\triangleright\) Inputs: \(u\)-current node, \(v\)-neighbor node, \(w\)-estimated travel time (cost) from \(u\) to \(v\), Returns: \(\pi[v]\)-predecessor of each node
18: if \(d[v] > d[u] + w(u, v)\) then
19: \(d[v] = d[u] + w(u, v)\)
20: \(\pi[v] = u\)
21: end if
22: end function

23: function MAIN(\(V, \varepsilon, Y, s\)) \(\triangleright\) Inputs: \(\forall\)-list of nodes, \(\varepsilon\)-list of edges, \(s\)-source node, \(Y\)-observation matrix, Returns: \(\pi[v]\)-predecessor of each node, \(w\)-edge weight matrix
24: \(P := NIL\)
25: \(Q := queue(\forall)\)
26: \(k := 0\)
27: \(p\in[s] = 0\)
28: \(w[p\in[s], s] = 0\ initialise\_single\_source(\forall, \varepsilon, s)\)
29: while \(Q! = 0\) do \(u := Extract\ min\_priority\ queue(Q)\)
30: \(p\in[u], npred = find\_prev(u, s) k = npred+1\ pW = w[p\in[u], u] P := P \cup u\)
31: for each \(v \in Adj[u]\) do
32: \(w[u, v] = find\_cost(u,v,k, pW,Y(k))\)
33: relax(u,v,w)
34: end for
35: end while
36: end function

The travel times of edges are identified as one of the cost coefficients in internal automated logistics. A formulation is devised to estimate travel times online during path computation considering its time-varying components. Moreover, suitable observations for travel time are recorded in scenarios with analogy to real factory in a scaled platform developed...
in the laboratory. They are instrumental for feeding into estimation algorithms to estimate travel time. Path is found using Dijkstra’s algorithm based on both heuristic weights of edges and estimated travel times of edges as weights. Results show that paths computed using travel time as weights of edges have lesser total path cost than that of obtained by heuristic weights. Environmental factors based costs are modeled as Gaussian process regression from already obtained finite measured data in [23], but it does not include time-varying changes in batteries and environment. On the other hand, sampling based heuristic path planning [7], [16] requires to explore a significant portion of the graph needs to find a suitable path, which is computationally expensive. Nevertheless, in this work, the cost of traversing every edge is estimated, which facilitates to apply deterministic path planning algorithms like Dijkstra’s algorithm, Bellmont-Ford algorithm et cetera. The approach used in single-task case in this work can be extended in multi-task scenarios for a MR, where cost coefficient for different tasks has to be found out. This is a direction for future consideration. During the run-time of MRS, every estimated value of travel time has context depending on various environmental and inherent factors. Travel time of one MR can provide contextual information to other MRs in an multi-robot system (MRS) and contribute in estimating travel time for them. This enhances further investigation towards implementing collaborative or collective intelligence in MRS.

REFERENCES

[1] Markus W Achtelik, Steven Weiss, Maria Chli, and Roland Siegwart. Path planning for motion dependent state estimation on micro aerial vehicles. In Robotics and Automation (ICRA), 2013 IEEE International Conference on, pages 3926–3932. IEEE, 2013.

[2] Maram Alajlan, Anis Koubba, Imen Chaari, Hachemi Bennaceur, and Adel Ammar. Global path planning for mobile robots in large-scale grid environments using genetic algorithms. In Individual and Collective Behaviors in Robotics (ICBR), 2013 International Conference on, pages 1–8. IEEE, 2013.

[3] Haluk Bayram and H Işıl Bozma. Coalition formation games for dynamic multirobot tasks. The International Journal of Robotics Research, 35(5):514–527, 2016.

[4] Patric Beinschob and Christoph Reinke. Graph slam based mapping for agv localization in large-scale warehouses. In Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on, pages 245–248. IEEE, 2015.

[5] Subhrabaj Bhattacharya, Maxim Likachev, and Vijay Kumar. Multi-agent path planning with multiple tasks and distance constraints. In Robotics and Automation (ICRA), 2010 IEEE International Conference on, pages 953–959. IEEE, 2010.

[6] Xiong Chen, Yingying Cong, Xiang Fang, and Qidi Wu. A fast two-stage aco algorithm for robotic path planning. Neural Computing and Applications, 22(2):313–319, 2013.

[7] Marcello Cirillo, Tansel Uras, and Sven Koenig. A lattice-based approach to multi-robot motion planning for non-holonomic vehicles. In Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, pages 232–239. IEEE, 2014.

[8] Mitchell Colby, Jen Jen Chang, and Kagan Tumer. Implicit adaptive multi-robot coordination in dynamic environments. In Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, pages 5168–5173. IEEE, 2015.

[9] Pragna Das and Lluís Ribas-Xirgo. Predicting battery level analysing the behaviour of mobile robot. In VIII Workshop of Physical Agents Book of Proceedings, pages 91–98, 2016.

[10] S Hamid Dezfoulian, Dan Wu, and Imran Shafiq Ahmad. A generalized neural network approach to mobile robot navigation and obstacle avoidance. In Intelligent Autonomous Systems 12, pages 25–42. Springer, 2013.

[11] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multi-robot coordination: A survey and analysis. Proceedings of the IEEE, 94(7):1257–1270, July 2006.

[12] Mohamed Elbanhawi and Milan Simic. Sampling-based robot motion planning: A review. IEEE Access, 2:56–77, 2014.

[13] Alessandro Farinelli, Luca Iocchi, and Daniele Nardi. Distributed online dynamic task assignment for multi-robot patrolling. Autonomous Robots, pages 1–25, 2016.

[14] B Gerkey and Maja J Mataric. Are (explicit) multi-robot coordination and multi-agent coordination really so different. In Proceedings of the AAAI spring symposium on bridging the multi-agent and multi-robotic research gap, pages 1–3, 2004.

[15] Thi Thoa Mac, Cosmin Copot, Duc Trung Tran, and Robin De Keyser. Heuristic approaches in robot path planning: A survey. Robotics and Autonomous Systems, 86:13–28, 2016.

[16] Neil Mathew, Stephen L Smith, and Steven L Waslander. Optimal path planning in cooperative heterogeneous multi-robot delivery systems. In Algorithmic Foundations of Robotics XI, pages 407–423. Springer, 2015.

[17] R. Parasuraman, K. Kershaw, P. Pagala, and M. Ferre. Model based on-line energy prediction system for semi-autonomous mobile robots. In 2014 5th International Conference on Intelligent Systems, Modelling and Simulation, pages 413–416, 2014.

[18] MB Priestley. Current developments in time series modelling. Journal of Econometrics, 37(1):67–86, 1988.

[19] Lluís Ribas-Xirgo and Ismael F Chaillé. An agent-based model of
autonomous automated-guided vehicles for internal transportation in automated laboratories. In ICAART (1), pages 262–268, 2013.

[20] Lluis Ribas-Xirgo, Joaquin Saiz-Alcaine, Jonatan Trullas-Ledesma, and A Josep Velasco-Gonzalez. An approach to a formalized design flow for embedded control systems of micro-robots. In Industrial Electronics, 2009. IECON’09. 35th Annual Conference of IEEE, pages 2187–2192. IEEE, 2009.

[21] Maria Isabel Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties, 2004.

[22] Indranil Saha, Rattanachai Ramaithitima, Vijay Kumar, George J Pappas, and Sanjit A Seshia. Implan: scalable incremental motion planning for multi-robot systems. In Cyber-Physical Systems (ICCPS), 2016 ACM/IEEE 7th International Conference on, pages 1–10. IEEE, 2016.

[23] Junghun Suh and Songhwai Oh. A cost-aware path planning algorithm for mobile robots. In Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, pages 4724–4729. IEEE, 2012.

[24] Yong Zhang, Dun-wei Gong, and Jian-hua Zhang. Robot path planning in uncertain environment using multi-objective particle swarm optimization. Neurocomputing, 103:172–185, 2013.