Decentralized Hash Tables For Mobile Robot Teams Solving Intra-Logistics Tasks

D. Sun
University of Freiburg
Georges-Köhler-Allee 52
79110 Freiburg, Germany
sun@informatik.uni-freiburg.de

A. Kleiner
University of Freiburg
Georges-Köhler-Allee 52
79110 Freiburg, Germany
kleiner@informatik.uni-freiburg.de

C. Schindelhauer
University of Freiburg
Georges-Köhler-Allee 51
79110 Freiburg, Germany
schindel@informatik.uni-freiburg.de

ABSTRACT

Although a remarkably high degree of automation has been reached in production and intra-logistics nowadays, human labor is still used for transportation using handcarts and forklifts. High labor cost and risk of injury are the undesirable consequences. Alternative approaches in automated warehouses are fixed installed conveyors installed either overhead or floor-based. The drawback of such solutions is the lack of flexibility, which is necessary when the production lines of the company change. Then, such an installation has to be re-built.

In this paper, we propose a novel approach of decentralized teams of autonomous robots performing intra-logistics tasks using distributed algorithms. Centralized solutions suffer from limited scalability and have a single point of failure. The task is to transport material between stations keeping the communication network structure intact and most importantly, to facilitate a fair distribution of robots among loading stations. Our approach is motivated by strategies from peer-to-peer-networks and mobile ad-hoc networks. In particular we use an adapted version of distributed heterogeneous hash tables (DHHT) for distributing the tasks and localized communication. Experimental results presented in this paper show that our method reaches a fair distribution of robots over loading stations.

Categories and Subject Descriptors

Autonomous vehicles [**Multiagent systems**]: Coherence and coordination

General Terms

Algorithms

Keywords

Distributed problem solving, Peer to peer coordination, Mobile agents, multi-robot systems, robot coordination

1. INTRODUCTION

Although a remarkably high degree of automation has been reached in production and intra-logistics nowadays, handcarts and forklifts manually steered by humans are still indispensable in many of situations. For example, boxes filled with small parts by a automated picking system have to be delivered to packing stations. Such tasks are still done by humans with handcarts, which is clearly undesirable due to the high costs and risk of failure. Alternatively, the problem is solved in automated warehouses with fixed installed conveyors either overhead- or floor-based. However, those solutions have several drawbacks, for example, when the business model of the company changes existing installations have to be redesigned.

Due to great advances of basic technologies, such as RFID, wireless networking, and robotics, several fully automated and flexible material flow handling systems have been introduced recently in the past. Examples of this include the Kiva MFS [20], and solutions within the framework of the "Internet of Things" [15]. However, these systems have three major disadvantages: First, they are all implemented by centralized control, thus having limited scalability with increasing number of robots and loading stations. Second, they require the user to engineer the environment. For example, in the Kiva MFS the entire floor has to be covered with Barcodes placed half a meter apart. Third, the system can only operate within human-free workspaces since there are neither sensors nor mechanisms for avoiding collisions with people.

The long-term vision behind the system introduced in this paper is to remedy these limitations, i.e., to build-up a team of autonomous and decentralized units that communicate on a low-range-basis with each other. In this system a team consisting of hundreds of robots will organize material flows autonomously and decentralized. Robots learn a map of the environment (which can be arbitrary) and localize themselves on this map. Furthermore, all units utilize autonomous path planning, plan execution, and collision avoidance, thus, facilitating safe control, also when human beings are around.

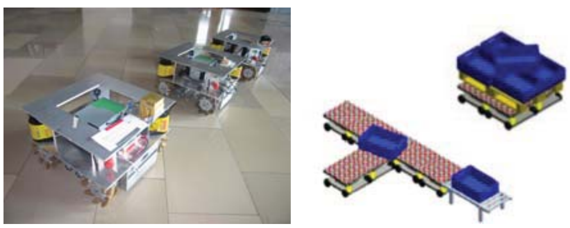
In this paper, we introduce a novel approach to coordinate autonomous robots performing intra-logistics tasks. Key challenges in this scenario are: to maintain transportation tasks within a mobile ad-hoc-network, keeping the network structure intact, and most importantly, to facilitate a fair distribution of robots among loading stations. The contribution of this paper is a solution to all of these problems at the same time by adopting a variant of the successful concepts used in modern peer-to-peer networks, namely Distributed Heterogeneous Hash Tables (DHHT). Note we do not handle the problem of multi-robot navigation, which could be solved, for example, by the plan merging paradigm introduced by Alami et al. [1]. Experimental results presented in this paper show that our method reaches a fair distribution of robots over loading stations.

Cite as: Decentralized Hash Tables For Mobile Robot Teams Solving Intra-Logistics Tasks , D. Sun, A. Kleiner, C. Schindelhauer, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 923-930

The reminder of this paper is organized as follows. In Section 2 we describe the hardware setting behind the *Karis* system utilized for experiments presented in this paper. In Section 3 distributed heterogeneous hash tables (DHHTs) are introduced and in Section 4 their integration for multi-robot team coordination is described. Finally, in Section 5 results from experiments, and in Section 6 our conclusions are presented.

2. THE KARIS SYSTEM

KARIS*(Kleinskalige Autonomes Redundantes Intralogistiksystem)*[7] is a small robot unit developed by a joint effort of the „Intralogistic Network“ in south Germany consisting of several companies and universities. The long-term goal of this project is to deploy hundreds of these elements to solve tasks in intra-logistics and production, such as autonomously organizing the material flow between stations. Figure 1 (a) depicts the third prototype of the KARIS element, which is equipped with a holonomic drive to facilitate docking behaviors. Missing on the picture is the conveyor for loading and unloading boxes when docked with a loading station which was not needed during our experiments. The element has a size of 50×50 cm, a payload of 60 kg, and is capable to recharge its batteries via contact-less rechargers let into the ground. Furthermore, it contains a high precision mechanism for enabling automatic docking maneuvers, either with other elements or a loading station. The docking mechanism enables the inter-connection of multiple KARIS elements as shown in Figure 1 (b). This has the advantage that multiple small elements can be combined to carry larger goods, such as pallets, or to temporarily form an assembly line.



(a) (b)

1: (a) The team of three KARIS robots used for experiments. (b) Inter-robot connections facilitated by a docking mechanism for jointly solving tasks.

For the purpose of autonomous navigation the element is equipped with two SICK S300 laser range finders (LRFs) mounted in two opposing corners, wheel odometry, and an inertial measurement unit (IMU). Navigation is based on A* [16] planning on grid maps which we generate from data collected by once steering a single robot manually through the environment. We use Monte-Carlo localization [5] with wheel odometry, IMU, and range readings from the two LRFs for localizing the robot on the grid map

3. DISTRIBUTED HETEROGENEOUS HASH TABLES

Distributed Heterogeneous Hash Tables (DHHT) were introduced as Weighted Distributed Hash Tables in [18] as an extension to Distributed Hash Tables (DHT) [10]. The objective is to assign robots $R_1, R_2, \ldots$ to a set of station $S_1, \ldots, S_n$ with weights $w_1, \ldots, w_n$ while preserving balance and consistency. In this paper we extend this approach to allow locality.

For this we consider a continuous hash range space $\mathcal{S}$ which in this paper constitutes the area used by the robots. The robots choose a uniformly distributed random (virtual) position in this space $\mathcal{S}$. Robot R_i is assigned to station S_j if j minimizes $\frac{||R_i - S_j||}{w_j}$, where $||R_i - S_j||$ denotes the distance between the position of station S_j and the starting position of R_i.

First note that this scheme preserves monotony and consistency [10] which means, that if a single station is inserted or the weight of a single station is increased, then robots will be assigned from other stations to this stations. No swapping of robots between other stations takes place. Similarly, if the weight of a station decreases or if it is removed, then only this station will free up robots for other stations.

For random station positions it is known that this scheme approximates the weighted balance, i.e. every station receives a share of $\frac{w_i}{\sum_j w_j}$ robots.

Theorem 1 *[18] The linear DHHT assigns with probability of at most* $\frac{w_i}{\sum_{j \neq i} w_j}$ *a robot to the station* S_i.

This result can be ameliorated by introducing a logarithmic distance measure. The logarithmic DHHT assigns a robot R_i to station S_j if j minimizes $\frac{\ln ||R_i - S_j||}{w_j}$ where the maximum possible distance in $\mathcal{S}$ is normalized to 1. For this measure an improved bound is known:

Theorem 2 *[18] The logarithmic DHHT assigns with probability of at most* $\frac{w_i}{\sum_j w_j}$ *a robot to the station* S_i.

Both methods suffer from the starting random choice of stations. Perfect balancing is only possible if multiple virtual random positions (copies) are used. However, using copies does not help in our case since we face non random positions of stations which we do not want to give up to preserve locality. By locality we mean that robots within the surrounding of the stations have higher chances to be assigned to the stations than farther robots.

If we allow the use of placeholder values $w'_1, \ldots, w'_n$ for the weights in the method which do not necessarily reflect the desired weighting $w_1, \ldots, w_n$ we can have all features: locality, balance, monotony and consistency.

Theorem 3 *For any placement of stations* $S_1, \ldots, S_n$ *and for all* $w_1, \ldots, w_n > 0$ *there exists a choice* $w'_1, \ldots, w'_n$ *for the linear and logarithmic DHHTs such that the probabilities for a given robot with random placement* $p_i(w'_1, \ldots, w'_n)$ *to be assigned to* S_i *is* $\frac{w_i}{\sum_j w_j}$.

Proof Sketch: For the proof we need the following monotony lemma:

Lemma 1 *For all* $\vec{w} = w_1, \ldots, w_n > 0$ *and all vectors* $\vec{e} \neq 0$ *which consists of* n *entries which are either* 1 *or* 0 *consider the functions of the linear and logarithmic DHHT with* $f_i(x) = p_i(w_1(e_1 x - x + 1), \ldots, w_n(e_n x - x + 1))$ *for* $x \in \mathbb{R}^+$.

Then all $f_i(x)$ *are monotonically increasing, continuous and partially differentiable where* $\lim_{x\to\infty} f_i(x) = 1$.

PROOF. Note that we consider functions like

$$f_i(x) = p_i(w_1, w_2 \cdot x, w_3 \cdot x, w_4) .$$

So, some arguments remain constants while others linearly increase with x.

The continuity follows by the continuity of the minimum function and the underlying metric for both, linear and logarithmic, methods.

For the monotonicity consider a point p in the space $\mathcal{S}$. Now if for some x if p is assigned to a station i where $e_i = 0$, then by increasing x some terms $\frac{||p-S_j||}{w_j x}$ become smaller than $\frac{||p-S_i||}{w_i}$. Now if p has been assigned to some station j with $e_j = 1$, then this assignment remains, since the inequality $\frac{||p-S_j||}{w_j x} < \frac{||p-S_k||}{w_k x}$ for the linear method or $\frac{\ln ||p-S_j||}{w_j x} < \frac{\ln ||p-S_k||}{w_k x}$ for the logarithmic method holds independently from x.

Continuity and the property of partially differentiability can be easily derived by the properties of the metric and the discrete events where the minimum changes.

For the limit note that for points which are not stations we observe for both measures that $\lim_{x\to\infty} \frac{||p-S_j||}{w_j x} = 0$ which implies the claim. $\square$

First note, that for the linear and logarithmic DHHT

$$\lim_{w\to 0} p_i(\underbrace{w,\ldots,w}_{j-1}, 1, \underbrace{w,\ldots,w}_{n-j}) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

So, the theorem follows by proving that the multivariate function $g_i = p_i(z_1,\ldots,z_n) - \frac{w_i}{\sum_j w_j}$ for given $w_1,\ldots,w_n$ has a root. Now this follows from the algebraic properties of the functions $f_1,\ldots,f_n$. $\blacksquare$

Note that this is only a proof of existence. It is yet unknown how such an optimal choice of weights $w'_1,\ldots,w'_n$ can be achieved besides using an iteration method which relies on the monotonicity of the functions.

Later on, we use an iterative, heuristic method to compute such weights. Furthermore, simulation shows that for non-random locations in two-dimensional environment the following version of logarithmic performs better. The upper logarithmic DHHT assigns a robot R_i is assigned to station S_j if j minimizes $\frac{\ln(1+||R_i-S_j||)}{w_j}$. Here, no normalization of the maximum possible distance in $\mathcal{S}$ is necessary which allows the use of this assignment schemes in dynamic environments. Note that Theorem 3 also holds for this method using an analogous proof.

4. SYSTEM DESIGN

The system is composed of several nodes with limited communication range, which either are mobile or stationary. Stationary nodes are directly interfacing loading stations that are offering transportation tasks. Those tasks are once inserted via a stationary node into the network where they are stored redundantly by a DHHT jointly maintained by all the nodes. In fact, nodes are building a mobile ad-hoc network in which communication takes place by a routing algorithm described in Section 4.1.

Transportation tasks are defined by a unique task ID, the source station ID where to load the box, and the destination station ID where to unload the box. The map of the environment and locations of stationary nodes are known in advance by each member of the network. Hence, transportation tasks can be executed independently by any mobile node by planning and executing a trajectory on the map. Tasks are negotiated similarly to the contract net protocol [19] between stationary and mobile nodes. However, in contrast to the conventional protocol, the dynamic assignment of mobile nodes to stations is guided by a weighted hashing function that facilitates a fair distribution of mobile nodes to stations (see Section 4.4).

Our approach is inspired by the early peer-to-peer network structure called content addressable networks (CANs) [14]. There, peers are virtually placed in a two-dimensional square and each position in the square is mapped to a peer. Each peer is responsible for a local rectangle and communicates only with the virtual neighbor. Data is assigned using a hash function to a position and thus assigned to a peer.

In this paper, tasks (located at fixed stations) need to be assigned to moving robots. These nodes continuously change their position in the network, and, on the other hand, loading stations and their vicinity are the only locations of interest since they produce and consume all information stored in the DHHT, namely transportation tasks. Therefore, we will refer in the following to the network as mobile content-addressable network (MCAN).

In MCANs we are interested in distributing robots among stations with true locations in two-dimensional space, rather than distributing data, e.g. MP3 files, among peers with virtual locations in n-dimensional space. Likewise as in CAN we are utilizing a distributed hash table for gaining a balanced distribution of resources as will be described in the following.

4.1 Decentralized Routing

In a MCAN nodes are at any time aware about their neighbors within the communication range, and thus they can directly communicate with them. Such a network without central infrastructure is commonly known as mobile ad-hoc network using geographic routing, a.k.a. position-based routing. It is a reactive multi-hop routing scheme where forwarding decisions are based on geographical positions of the nodes in the network (see [12, 6] for surveys). It is based on the assumptions that all nodes know their positions and the position of the destinations.

In geographic routing only stationary nodes are directly addressable since only their locations are known a-priori. Mobile nodes can be addressed if they provide their current location beforehand within a request message.

In MCAN, we use a simplified variant of the greedy perimeter forwarding scheme routing [11]. Each node greedily forwards messages to its neighbor with shortest distance to the destination of the message. If a message cannot be delivered as a fall-back strategy the message is flooded using a backtracking mechanism. In contrast to other geographic approaches we do not thin out the communication links in order to maintain a planar network. This is usually necessary for fall-back strategies based on the right-hand rule. This is also motivated by the fact that our nodes move (or even crash) and we are trying to get the proactive communication overhead at a minimum. Since this dynamic network behavior can easily partition the network we cache copies of the messages and use the mobility of the nodes to transport messages. This technique is known as delay tolerant routing [9]. A more detailed description for detecting and repairing communication problems will be described in Section 4.3.

Routing efficiency is improved by adapting the routing metric (Euclidean space) according to the signal strength. For this, nodes with larger signal strengths are preferred from the set of nodes which decrease the distance to the target. So, more reliable connections are preferred while the consequence might be longer routes. In CAN the average path length has been estimated by $O(n^{1/2})$, in two-dimensional space and for n peers. In MCAN this might vary since the robots are not evenly distributed in a plane, e.g. for the communication network in a long aisle the maximum hop length will be of linear size. This does not really state a problem since the size and number of our messages is independent from the number of tasks in the queue and can be limited to a local surrounding of a base station. Furthermore, status information is spread in the network such that additional mobility is induced when necessary. As a side effect of the mobility, the (long-time) receiving range of a node is much larger than those of static nodes.

4.2 Station Status Information

Task allocation in MCAN relies solely on the up-to-date availability of the station status information (SSI) of all stations at the robots. Based on this information alone each robot decides to perform a task at a station. This information is updated in the network using beacon messages. All stations send periodically a station status information (SSI). The mobile nodes compute from this values and the current position the weighted hash key and determine the next contractor. Note that SSIs do not need to be broadcasted into the entire network but only into regions where nodes reside where the SSI may have an influence on the outcome of the DHHT. This technique has the potential to reduce the overhead of beacon messages to a constant amount for large networks. Since no link state information needs to be transformed and no information about specific tasks MCAN has minimal proactive traffic.

The SSI consists of the tuple $\langle t, N_Q, N_C, N_{delivered}, e\rangle$, where t is a time stamp indicating the freshness of the information, N_Q the current queue length, N_C the number of robots currently assigned to tasks in the queue, $N_{delivered}$ the total number of boxes delivered, and e a value expressing the station's current efficiency. Note that $N_{delivered}$ has to be bounded by a time window. In our implementation we considered the total number of delivered boxes within the last 20 minutes.

We define the *throughput rate* Tr as the number of tasks currently dispatched per time unit by a station. The maximal throughput rate Tr_{max} is given by the minimum between the number of boxes arriving at the loading station per time unit (which has to be sampled), and a constant k, where $1/k$ characterizes the dispatching latency, i.e. the time needed by a robot docking with the loading station, loading the box, and leaving it. Note that k can be different for each station, for example, due to physical constraints. The *efficiency* of station i is then defined by:

$$e_i = \frac{Tr}{Tr_{max}}. \quad (1)$$

Therefore, efficiency values close to 1.0 are typical for stations that are sufficiently visited by robots, whereas values close to 0.0 are representing stations that have been strongly unattended by the team. We have defined the efficiency to be non-zero because we are using this term as divisor later on. We define $e_i = 1.0$ if $Tr_{max} = 0$, and $e_i = \epsilon$ if $Tr = 0$, where ϵ is a positive constant close to zero.

4.3 MCAN Construction and Maintenance

When a robot is added to MCAN it performs the following three-step procedure for becoming a member of the network and becoming assigned to its first task:

1. **Search for the network.** The robot explores the physical space until it meets the communication range of any other robot connected with the network.
2. **Receive station SSIs.** The robot waits and collects for a certain time station status messages as described in Section 4.2.
3. **Station selection.** Based on the received station SSIs, the robot selects a station as a contracting partner according to the weighted hashing function, which will be explained in Section 4.4.

The fact that robots initially know the map of the environment, and also the list of existing stations, facilitates a simple mechanism for repairing the network. Network repairs are necessary in situations where the network is divided into subnets, and also when it is entirely disconnected. The latter is particularly the case during bootstraping, e.g., when the first robot enters the environment.

The detection of disconnected network components can be inferred by time stamps t_i and thus the age of all SSIs that are continuously sent by the stations. Note that we define the age of a SSI as infinite if it has never been received. If a mobile node receives an incomplete set of SSIs, or even an empty set, it establishes a connection between the network and a randomly selected station from this set by moving into the station's direction. Consequently, new SSIs will populate the network after the connection has been (re-)established.

4.4 DHHT-based task distribution

Mobile nodes that are initially placed into the network or just finished a previous task, have to be (re)assigned for new tasks. This is carried out by, first, deciding for a station based on the received SSIs, second, sending a task request message REQ to the selected station, and third, moving towards the station and negotiating for a specific task when within direct communication range. These messages are forwarded using geographic routing and constitute the reactive part of MCAN's routing protocol. Note that in contrast to other reactive MANET routing protocols like DSR (Dynamic Source Routing) or AODV (Ad-Hoc On Demand Routing) no routing information needs to be stored at the nodes nor within the message headers. Furthermore, no acknowledgment of the station is expected. To increase the reliability of this protocol a duplicate REQ is sent half way after sending the first REQ.

We will now describe the computation of the hash key $D_r(r_x, r_y, SSI_1, SSI_2, ..., SSI_n)$, which is computed with respect to the robot's current position (r_x, r_y), and latest SSIs received from the stations. From the SSIs, we computed for each station i a weighting factor w_i expressing the station's eligibility for mobile nodes:

$$w_i = \frac{1}{e_i}\frac{N_Q - N_C}{N_{delivered}}. \quad (2)$$

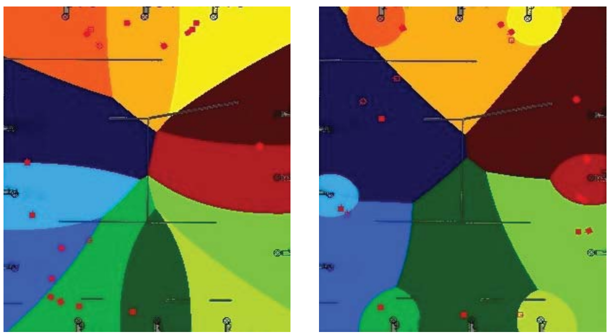
In Equation 2 the long-term demand $1/e_i$, but also the current queue length are mainly influencing the weighting. Furthermore, we shrink the real queue length by the current amount of robots N_C that already assigned for the station. This is important in order to avoid oscillations during runtime, i.e., situations in which robots are continuously swapped between stations. Note that station's are immediately aware of robots that assign to them due to the REQ message confirming their interest. Consequently, they will increase N_C leading automatically to a different weighting and thus less eligibility when compared to other stations.

The weighted hash key is finally computed for each station by:

$$D_i = \frac{\log(d_i)}{w_i}, \quad for \quad d_i > d_{min} \quad (3)$$

where w_i denotes the station's weighting, d_i the distance between robot and station, and d_{min} a minimum radius around the station which should be larger than 1. The d_i can straight forwardly be computed by the Euclidean distance. However, within confined spaces this distance metric is an inaccurate estimate since locations can arbitrarily be disconnected by walls. Therefore, we compute the d_i based on the true traveling distance, i.e., the length of the path a robot would travel between both locations, by a Voronoi Graph [4].

Finally, mobile nodes are assigning themselves to the station with minimal weighted distance D_i. When arriving in the vicinity of this station a new task is negotiated according to the contract net protocol [19], and after dispatching this task, the station decreases N_C and N_Q by one.



(a) (b)

2: Visualization of $min(D_i)$ for two different situations on the *small map*. Weighted distances D_i of each station i are depicted by a specific color. For each location the minimum D_i is drawn.

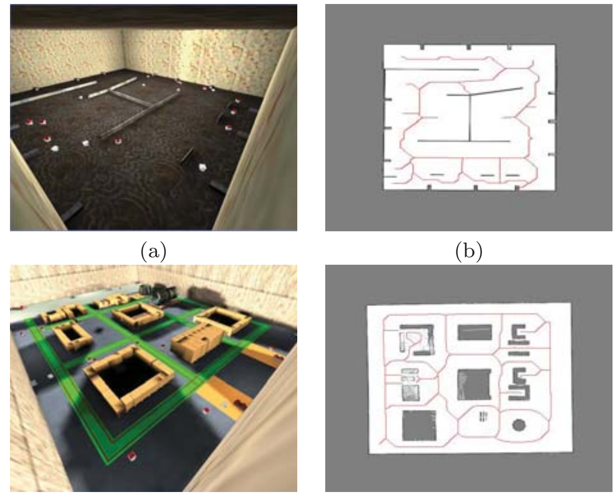
Figure 2 visualizes the effect of Equation 3 in two situations. Each station governs a different region for assigning mobile nodes. In the specific situation shown in (a) all stations were simulated with a moderate throughput rate of 4 boxes per minute, whereas in (b) 6 stations were simulated with a high throughput rate (5 boxes per minute), and 6 others with a low throughput rate (2 boxes per minute). Consequently, in (a) station areas are evenly distributed, whereas in (b) some stations have more influence than others.

5. EXPERIMENTS

The proposed approach has been evaluated with robot teams both in simulation and in a real-world scenarios. We tested the system according to its convergence behavior towards a fair robot distribution among the stations, and the systems stability with regards to rapid changes, such as a change of the workload of each station, and the situation that some robots suddenly fail.

5.1 Simulation Results

We conducted experiments with larger robot teams in test worlds simulated by the *USARSim* [3] framework. USARSim is an extension based on the game engine of the computer game *Unreal Tournament 2004*, and is particularly tailored for simulating heterogenous robot teams in the context of urban search and rescue (USAR) and automation. The game engine of UT2004 is utilized for simulating robot navigation and sensor perception in 3D. Hence, algorithms controlling robots in USARSim are processing the same kind and amount of data as they would do on real systems. For



(a) (b) (c) (d)

3: (a,c) Simulated environments in USARSim. (b,d) Grid maps for localization and planning with superimposed Voronoi graph (red). Maps are denoted as (a) *small factory*, and (c) *big factory*.

example, single agents receive in real-time via a TCP/IP interface all measured beams of their virtual laser range finders, and have to return within an appropriate time frame motor commands, e.g. translational and rotational velocities. Several authors validated the simulator with respect to its reality-closeness [2, 17, 8]. Experiments were conducted by running simultaneously two 8-core Intel Xeon (2.66GHz), two Dual-core Intel Xeon (3.06GHz), one Dual-core (2GHz), and one Intel Core2Duo (3GHz) CPUs.

Figure 3 (a,c) depicts some of the environments with virtual loading stations (small cuboids) and robots used for testing our approach. Figure 3 (b,d) depicts grid maps generated from these environments with superimposed Voroni graph utilized for the distance metric. We additionally used a map created according to the layout of one of the production halls from a larger company (not shown in the figure), which we denote as *Company map* in the following. For our experiments we used the wireless simulation server (WSS) [13] that simulates signal path attenuation of wireless network transmission between the peers. Therefore, robots that are far from each other or separated by walls experienced signal loss, i.e., had no possibility to communicate directly.

The arrival of boxes at loading stations was modeled by a Poisson distribution. Each station is characterized by the average number of packages λ_i arriving within one time unit. During each time step of the simulation, the amount k_i of packages generated at station i was stochastically drawn from:

$$P_{\lambda_i}(k_i) = \frac{\lambda_i^{k_i}}{k_i!} e^{-\lambda_i}, \tag{4}$$

Likewise to an assembly lines in the real world, arriving boxes were kept in a virtual queue attached to each station. Robots docking with the station automatically loaded the oldest box from the queue and delivered it to its destination. The destinations of boxes were drawn stochastically from a Gaussian distribution with mean $\mu = (s_x, s_y)$ according to the station's location, i.e., stations close to the source were chosen with higher probability than others. This mimics situations in a real plant where correlated production units are located more likely close to each other. When

1: Simulation Results WHHT & Baseline

	Small map	Big map	Company map
Base eff. [%]	0.64± 0.27	0.41 ± 0.27	0.8±0.11
WHHT eff. [%]	1.0± 0.02	0.64 ± 0.03	1.0±0.0
Base deliv. [#]	115.2± 44.0	77.4 ± 46.0	155±22.2
WHHT deliv. [#]	178.6± 1.6	151.4 ± 9.6	177.1±1.2
Base w. time [min.]	71.6	50.1	75.6
WHHT w. time [min.]	3.3	16.2	2.4

not stated differently, we used $\lambda = 4$ *boxes/min* during our experiments.

5.1.1 *Efficiency and Convergence*

We compared the proposed approach of WHHT-based robot distribution against the straight forward solution that we will denote by *baseline*. The baseline approach is to assign robots according to their spacial distance to stations. Robots receive task offers from all stations and decide for the station with the shortest distance. Note that the baseline approach does not automatically lead to the situation where robots only work for a subset of stations. After some time they might reach any station when accomplishing delivery tasks with far destinations. However, it is not guaranteed that they assign themselves evenly to the stations. Some stations can be disadvantaged due to their location.

Table 1 shows the impact of WHHT-based assignment on distribution efficiency and fairness. Whereas the average efficiency of all stations with baseline (Base eff.) is 0.64, 0.41, and 0.8, WHHT reached both higher total efficiency (the average values 1.0, 0.64, and 1.0), and a better degree of fairness (the standard deviations 0.02, 0.03, and 0.0). Also the total number of delivered packages (deliv.), and the worst-case time-delay of box deliveries (w. time) is much better in comparison to the baseline. Figures 4 (small map) and 5 (Company map) show the efficiency of each station over time. As can clearly be seen, when running with our method (b) the set of stations converges to a fair distribution of efficiency. Note that efficiencies above 1.0 during the beginning are due to queues that filled up while the robot team was still bootstrapping.

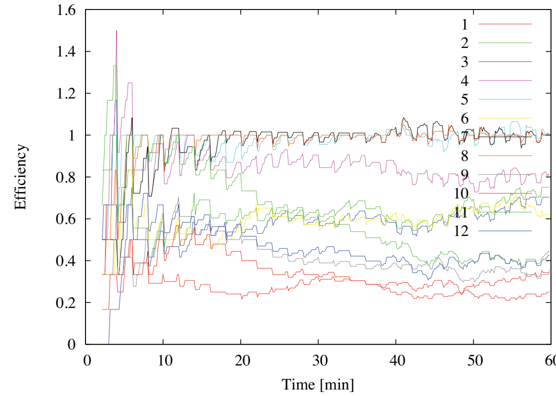
5.1.2 *Adaption to sudden changes*

Within another series of experiments we evaluated the behaviour of our approach in case of sudden changes of the environment. Note that all experiments were repeated several times. Figure 6 depicts the efficiency of stations over time, where the load at each station was suddenly changed after 30 minutes. Instead of $\lambda = 4$ we changed for 6 stations to $\lambda = 5$, and for the other 6 stations to $\lambda = 2$. The result shows that after the change the standard deviation of the average efficiency increased for some while, and then converged back to an acceptable level. Then, stations were again served in a fair manner, as enforced by our method.

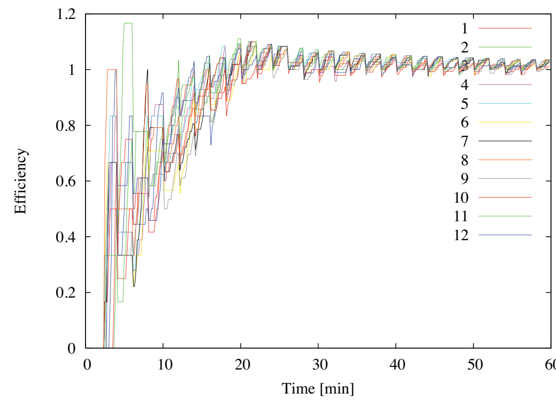
Figure 7 depicts the result from an experiment where the number of robots has been suddenly decreased. We run 18 robots on the *small map* and simulated a robot crash by removing 10 of them after 30 minutes. The figure shows that the team did react on the change, i.e., all stations were still continuously visited by robots. The overall efficiency (mean) and the overall fairness (standard deviation) decreased due to the significant change. Obviously, the performance has to decrease since a reduced set of robots is forced to travel larger distances on the map in order to serve all stations.

5.2 Real-World Results

Finally, we conducted real-world experiments with three KARIS elements. Stations were simulated in the same way as described in Section 5.1. During all experiments the



(a)



(b)

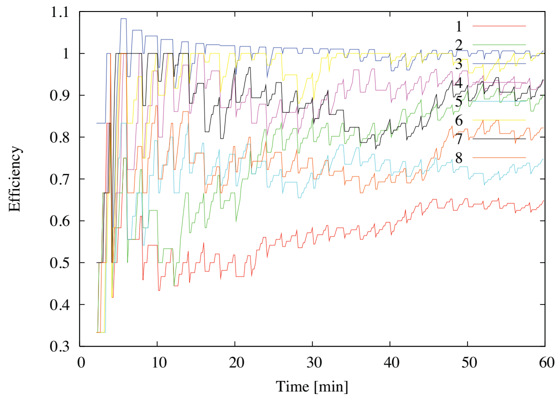
4: Efficiency of all 12 stations over time on the *small map*. Each color denotes a single station. (a) Baseline approach, and (b) WHHT approach.

2: Real-World Results WHHT & Baseline

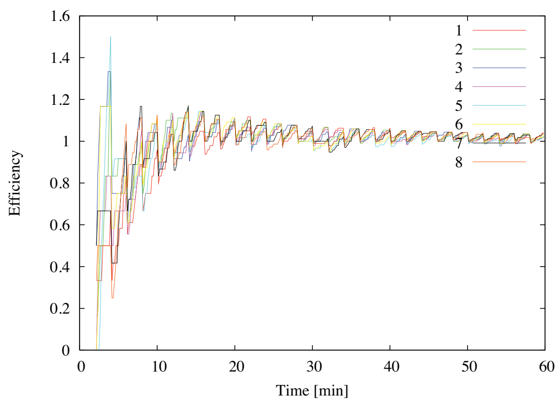
	Three stations	Four stations
Base eff. [%]	0.69± 0.1	0.74 ± 0.24
WHHT eff. [%]	0.76± 0.05	0.9 ± 0.09
Base deliv. [#]	36.3± 4.9	20.3 ± 6.5
WHHT deliv. [#]	24 ± 1.6	25.5 ± 2.7
Base w. time [min.]	20.1	18.3
WHHT w. time [min.]	14.5	10.5

robots were running in total more than ten hours autonomously without causing collisions or dead-locks. They finished transportation tasks for up to four stations, where one station was placed comparably far from the others. Figure 8 depicts the map of the testing environment with stations marked as circles. The coloring shows the weighted distance computed for each station. In the particular moment shown by the figure, station 1, located on the far left end, has the strongest influence (dark orange) on the robots due to the WHHT that automatically compensates the station's outer location.

Table 2 summarizes the results from the experiments with three stations ($\lambda = 2$), and with four stations ($\lambda = 3.5$), respectively. As can be seen, the WHHT assignment reaches a more fair distribution (standard deviation), better efficiency, and a smaller worst-case time of deliveries. Figure 9 depicts the efficiency over time, which is also here more evenly distributed when applying the WHHT. We also noticed during real-world experiments that the system appropriately reacted to changes, i.e., redistributed its resources. For example, in one situation a robot was running out of batteries,

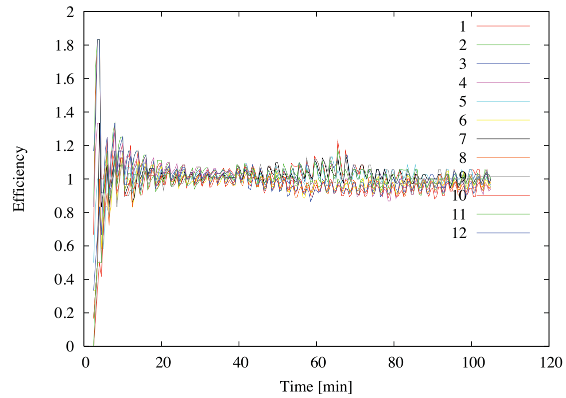


(a)

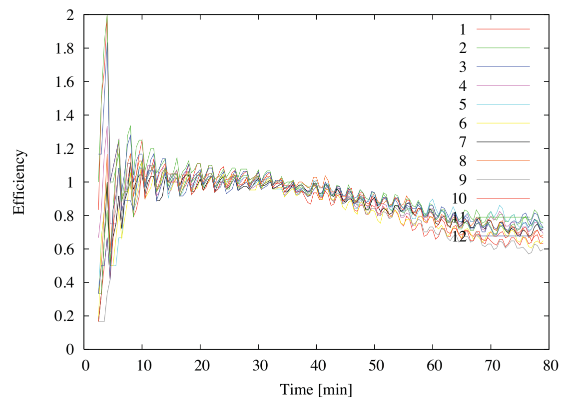


(b)

5: Efficiency of all 8 stations over time on the *Company map*. Each color denotes a single station. (a) Baseline approach, and (b) WHHT approach.



6: Adaption after rapid throughput change. For the first 30 minutes all 12 station were running at $\lambda = 4$, and from then on 6 with with $\lambda = 5$ and 6 with $\lambda = 2$



7: Adaption after a sudden break-down. For the first 30 minutes the system was running with 18 robots, then, we intentionally crashed 10 of them.

and in another one we increased the travel time to one station by blocking the direct path.

6. CONCLUSION

We have proposed a novel approach for a decentralized team of autonomous robots performing intro-logistics tasks. Our approach is decentralized and does not use any additional hardware infrastructures which allows the fast deployment in any production plant. For location and navigation it uses laser range finders, mounted in two opposing corners, wheel odometry, and an inertial measurement unit location, while competing approaches rely on extra built-in devices. For communication we propose a mobile ad-hoc network which consists of a hybrid approach of a proactive station status dissemination protocol and a reactive geographic routing scheme for task requests by the robots. Other protocols require the installment of WLAN routers or field buses. In our approach only the loading stations and the robots carry communication devices.

Furthermore, there is no central server which coordinates the communication or the task allocations. In fact it is not necessary to connect the loading stations to a local area network. All necessary communication is performed by the robots and the loading stations. For the allocation process only robots decide autonomously which station is the next to be served. It is done by an adapted version of Distributed Heterogeneous Hash Tables (DHHT) known from peer-to-peer networks. Such communication networks constitute a robust alternative for client-server networks.

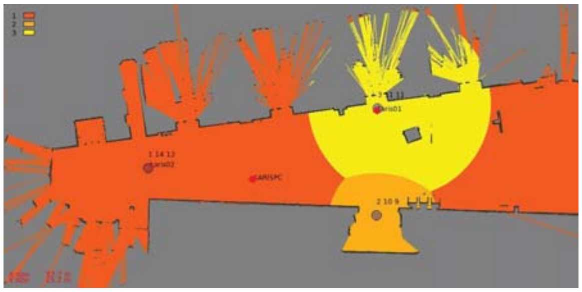
Because of the robustness of all system components, the distributed algorithms and the autonomy of all devices the resulting system, MCAN, is able to cope with dynamics, partial failures and is highly scalable. Failures of robots or adding further robots can happen during the run-time barely influencing the system. Furthermore, it easily adapts to changing demands and provides a fair treatment of the loading stations.

An additional feature is the user transparency of the task allocation mechanism which provides a locality feature. Each loading station is responsible for a dynamically changing catchment area. These geometric areas adapt to the demand and queue length of the stations. So, a graphical user interface gives the users a pictorial description which demonstrates the decision process of each individual robot and gives the user a good intuition of future assignments.

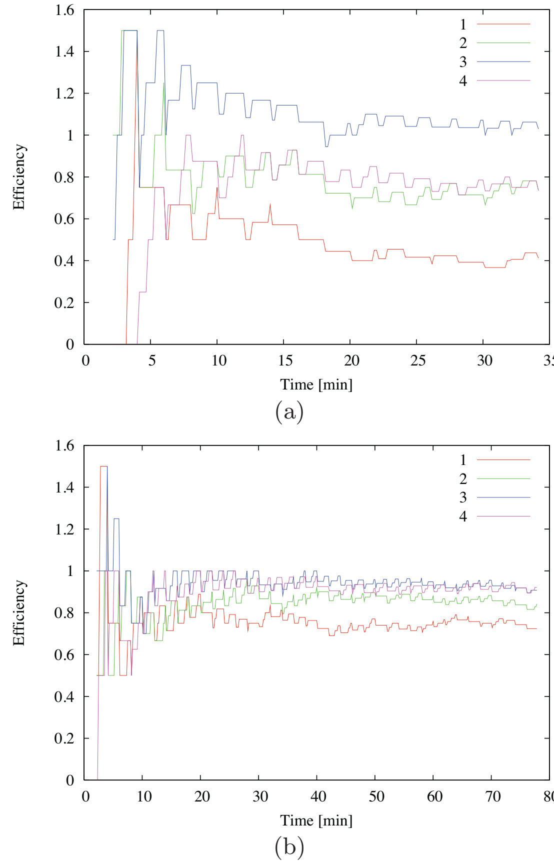
Besides the theoretical findings for the allocation mechanism, we have performed several simulations and a real-world experiments with a small group of three robots testing the allocation successfully under real-world constraints.

7. REFERENCES

[1] R. Alami, F. Robert, F. Ingrand, and S. Suzuki. Multi-robot cooperation through incremental plan-merging. In *Proc. of the IEEE Int. Conf. on*



8: Map of the real-world scenario and visualization of $min(D_i)$. The far-off station on the left side (dark orange) has the highest demand for robots.



9: Efficiency of all 4 stations over time in the real environment. (a) Baseline approach, and (b) WHHT approach.

Robotics & Automation (ICRA), volume 3, pages 2573–2579, May 1995.

[2] B. Balaguer, S. Balakirsky, S. Carpin, M. Lewis, and C. Scrapper. USARSim: a validated simulator for research in robotics and automation. In *Workshop on "Robot Simulators: Available Software, Scientific Applications, and Future Trends" at IEEE/RSJ*, 2008.

[3] S. Carpin, M. Lewis, J. Wang, S. Balakirsky, and C. Scrapper. USARSim: a robot simulator for research and education. In *Proc. of the 2007 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1400–1405, Roma, Italy, 2007. Citeseer.

[4] H. Choset, , and Burdick J. Sensor-based exploration: The hierarchical generalized voronoi graph. *Int. Journal of Robotics Research*, 19(2), 2000.

[5] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1998.

[6] Silvia Giordano and Ivan Stojmenovic. Position-based ad hoc routes in ad hoc networks. In Mohammad Ilyas, editor, *The Handbook of Ad Hoc Wireless Networks*, chapter 16. CRC Press, 2003.

[7] H. Hippenmeyer, K. Furmans, T. Stoll, and F. Schönung. Ein neuartiges Element für zukünftige Materialflusssysteme. *Hebezeuge Fördermittel: Fachzeitschrift für Technische Logistik*, (6), 2009.

[8] S. Hughes M. Koes J. Wang, M. Lewis and S. Carpin. Validating usarsim for use in hri research. In *Proceedings of the Human Factors and Ergonomics Society 49th Annual Meeting*, pages 457–461, 2005.

[9] S. Jain, K. R. Fall, and R. K. Patra. Routing in a delay tolerant network. In Raj Yavatkar, Ellen W. Zegura, and Jennifer Rexford, editors, *SIGCOMM*, pages 145–158. ACM, 2004.

[10] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 654–663, El Paso, Texas, 1997.

[11] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 243–254, 2000.

[12] M. Mauve, J. Widmer, and H. Hartenstein. A survey on position-based routing in mobile ad hoc networks. *IEEE Network Magazine*, 15(6):30–39, November 2001.

[13] M. Pfingsthorn. Wireless simulation server – Documentation, 2008. Available at: `http://robotics.jacobs-university.de/VirtualRobots/WSS.pdf`.

[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Proceedings of the 2001 SIGCOMM conference*, volume 31, pages 161–172. ACM New York, NY, USA, 2001.

[15] M. Roidl, L. Nagel, and G. Follert. The internet of things: On standardisation in the domain of intralogistics. In Jakobs, editor, *Workshop Producing Standards for the Internet of Things*, 2008.

[16] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.

[17] Y. Nevatia M. Lewis J. Wang S. Carpin, T. Stoyanov. Quantitative assessments of USARSim accuracy. In *Proceedings of PerMIS*, 2006.

[18] C. Schindelhauer and G. Schomaker. Weighted distributed hash tables. In *17th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 218–227, 2005.

[19] Reid G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C.29, 1980.

[20] P.R. Wurman, R. D Andrea, and M. Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Magazine*, 29(1):9–20, 2008.