The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Control of ad-hoc formations for autonomous airport snow shoveling

Martin Saska and Vojtěch Vonásek and Libor Přeučil

Abstract— **In this paper, we present a framework that applies multiple groups of autonomous snowploughs for efficiently removing the snow from airfields. The proposed approach includes formation stabilization into variable shapes depending on the width of runways. The paper is focused on trajectory planning and control during splitting and coupling of formations for cleaning smaller auxiliary roads surrounding main runways. We propose a general method using a receding horizon control for iterative formation assignment. The algorithm is adapted for the kinematics of car-like robots and can be utilized in arbitrary static and dynamic airport assemblage. The proposed approach has been verified by simulations and by hardware experiments.**

I. Introduction

The main task of the ground staff at the airport is to maintain airports' operations safe and uninterrupted. One of the critical periods for the safeness of air traffic is snowy weather. It is a complicated task to remove the snow from the huge surface of the main runways and the big amount of auxiliary roads that are necessary for the planes as well as for the ground vehicles. Due to the fact that the big airports are already equipped by the essential sensors, i.e. a global positioning system and automatic detection of runway conditions, an autonomous cleaning system can be set up relatively easily. Also the periodicity of the task predestinates the use of autonomous robots.

The basic idea of the proposed system is motivated by current approaches commonly used for shoveling of runways by human driven snowploughs. Since partly cleaned paths could be dangerous especially in emergency situations, it is required that the main runways as well as the auxiliary roads have to be cleaned up at once. Thus we avoid forced landing planes as well as rescue and fire fighting vehicles facing roads with an irregular snow surface. The main runway should be therefore cleaned up by a big group with sufficient numbers of vehicles. When the cleaning of the runway is accomplished the big group is splitted into smaller ad-hoc teams with sizes appropriate for the shoveling of smaller roads. Such an approach requires an algorithm for control of formations of autonomous ploughs during the group assemblage, which is the aim of this paper.

The snow shoveling task addressed in this paper is related to the field of autonomous sweeping. An initial work of cooperative sweeping by multiple mobile robots has been published in [5]. The application of this method in a real environment has been enabled by approach in [6]. Another example of decentralized cleaning algorithm is a method inspired by cooperative behavior of ants introduced in [10]. Projecting the problem of cooperative sweeping onto the airfield, the snow shoveling task generates some problems that have not yet been solved. Our approach considers the nonholonomic kinematics of usual snowploughs as well as the position and orientation of the ploughs' shovels. Furthermore, the working space is more structured due to the airfield environment.

As was mentioned above, the safety requirements at the airport predestinates the utilization of autonomous formations, which has not yet been investigated in terms of sweeping. In the classical literature, formation driving approaches are divided into the three main groups: virtual structure, behavioral techniques, and leader-follower methods [9]. We have chosen the leader-follower method as an appropriate approach for maintaining of ploughs with car-like kinematics. In this algorithm the followers maintain their position in the formation relative to the leader and therefore the state of the leading vehicle needs to be distributed within the team. In the literature, there is a broad offer of methods for formation stabilization in desired positions [2] as well as for driving along predefined trajectories [1], but approaches solving continues splitting and merging of independent groups have not yet been investigated. We designed such a framework solving the assemblage of ad-hoc formations in the dynamic environment of airports. The method is enough robust to deal with unforseen obstacles, changes in map structure as well as with failures of ploughs.

II. System overview

The structure of the introduced airport cleaning system is based on a central supervision for the high level coordination. The highest level of the proposed scheme (see Fig. 1) is divided into two types of units. The core of the first one (denoted *Command Center*) is the *Task Allocation* module, which designs temporary collations for independent tasks. A task can be for example to clean a runway or road. The second kind of units (blocks denoted as *Formation I* - *Formation* $n_{\mathcal{F}}$) represents the current constellation of vehicles where each unit corresponds to one formation. These units, which are independent from the *Command Center* most of the time, are primarily responsible for putting the assigned task into the appropriate formation motion.

Here, the module *Leader* is responsible for generating a reference trajectory for the complete formation at the beginning of each task received from the *Task Allocation* module. Physically, the module *Leader* is a routine launched on an internal PC of a designated robot. Besides the snow shoveling

Authors are with Department of Cybernetics, Czech Technical University in Prague. {saska,vonasek,preucil}@labe.felk.cvut.cz

This work is supported by CTU under the grant SGS10/195/OHK3/2T/13 and by European Union under the grant Symbrion-Enlarged no. 258273.

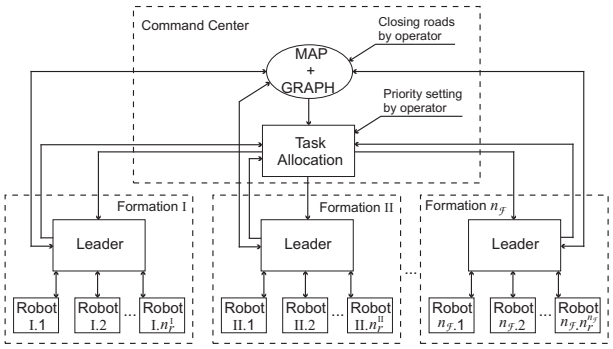


Fig. 1. Scheme of the complete snow shoveling system. The arrows denote communication links between the different modules.

such a robot acts as a connection between the formation and the *Command Center*. It informs the *Command Center* about detected obstacles, finished or aborted tasks as well as about the need of additional robots to compensate failures.

Note that the reference point for the formation movement is called the virtual leader in this paper and its position can be situated independently from the positions of the ploughs. The individual control inputs are calculated separately by each follower in order to follow the reference trajectory while maintaining the formation. This paper is focused on description of the second part responsible for the formation control while a comprehensive study of the first part, task assignment, and a more detailed description of the airport snow shoveling project can be found in our previous publications [3], [7].

III. Preliminaries

Let $\psi_j(t) = \{x_j(t), y_j(t), \theta_j(t)\} \in \mathcal{C}$, where $j \in \{1, \ldots, n_r, L\}$, denote the configuration of each of the n_r followers and a virtual leader L of formation $\mathcal{F}$ at time t. The Cartesian coordinates $x_j(t)$ and $y_j(t)$ define the position $\bar{p}_j(t)$ of a robot R_j and $\theta_j(t)$ denotes its heading. Let us assume that the environment of the robots contains a finite number n_0 of compact obstacles collected in a set of regions $\mathcal{O}_{obs}$. Finally, we need to define a circular detection boundary with radius r_s and a circular avoidance boundary with radius r_a, where $r_s > r_a$. Single robots should not respond to obstacles detected outside the region with radius r_s. On the contrary, distance between the robots and obstacles less than r_a is considered as inadmissable.

The kinematics for any robots R_j, where $j \in \{1, \ldots, n_r, L\}$, is described by the simple nonholonomic kinematic model: $\dot{x}_j(t) = v_j(t)\cos\theta_j(t)$, $\dot{y}_j(t) = v_j(t)\sin\theta_j(t)$ and $\dot{\theta}_j(t) = K_j(t)v_j(t)$. Velocity $v_j(t)$ and curvature $K_j(t)$ represent control inputs $\bar{u}_j(t) = \{v_j(t), K_j(t)\} \in \mathbb{R}^2$.

Let us define a time interval $\langle t_0, t_N\rangle$ containing a finite sequence with N elements of nondecreasing times $\mathcal{T}(t_0) := \{t_0, t_1, \ldots, t_{N-1}, t_N\}$, such that $t_0 < t_1 < \ldots < t_{N-1} < t_N$. Also, let us define a controller for a robot R_j starting from a configuration $\psi_j(t_0)$ by $\mathcal{U}_j(t_0) := \{\bar{u}_j(t_0; t_1 - t_0), \bar{u}_j(t_1; t_2 - t_1), \ldots, \bar{u}_j(t_{N-1}; t_N - t_{N-1})\}$. Each element $\bar{u}_j(t_k;\, t_{k+1} - t_k)$, where $k \in \{0, \ldots, N-1\}$, of the finite sequence $\mathcal{U}_j(t_0)$ will be held constant during the time interval $\langle t_k, t_{k+1}\rangle$ with uniform length denoted as Δt.

Let us integrate the kinematic model over a given interval $\langle t_0, t_N\rangle$ with constant control inputs from $\mathcal{U}_j(t_0)$ in each time interval $\langle t_k, t_{k+1}\rangle$, where $k \in \{0, 1, \ldots, N-1\}$ (from this point we may refer to t_k using its index k). By this integrating, we can obtain the following model for the *transition points* at which control inputs change:

$$
\begin{aligned}
x_j(k+1) &= \begin{cases} x_j(k) + \frac{1}{K_j(k+1)}\left[\sin\left(\theta_j(k) + \right.\right. \\ \left. K_j(k+1)v_j(k+1)\Delta t\right) - \\ \left.\sin\left(\theta_j(k)\right)\right], \text{if } K_j(k+1) \neq 0; \\ x_j(k) + v_j(k+1)\cos\left(\theta_j(k)\right)\Delta t, \\ \text{if } K_j(k+1) = 0 \end{cases} \\
y_j(k+1) &= \begin{cases} y_j(k) - \frac{1}{K_j(k+1)}\left[\cos\left(\theta_j(k) + \right.\right. \\ \left. K_j(k+1)v_j(k+1)\Delta t\right) - \\ \left.\cos\left(\theta_j(k)\right)\right], \text{if } K_j(k+1) \neq 0; \\ y_j(k) + v_j(k+1)\sin\left(\theta_j(k)\right)\Delta t, \\ \text{if } K_j(k+1) = 0 \end{cases} \\
\theta_j(k+1) &= \theta_j(k) + K_j(k+1)v_j(k+1)\Delta t,
\end{aligned}
\tag{1}
$$

where $\psi_j(k) = \{x_j(k), y_j(k), \theta_j(k)\}$ is the configuration at the transition point with index k. Control inputs $v_j(k+1)$ and $K_j(k+1)$ are extracted from $\bar{u}_j(k+1) := \bar{u}_j(t_k; t_{k+1} - t_k)$ at time index $k+1$.

In applications, the control inputs are limited by vehicle mechanical capabilities (i.e., chassis and engine). These constraints can be taken into account for each robot R_j limiting their control inputs by the following inequalities: $v_{min,j} \leq v_j(k) \leq v_{max,j}$ and $|K_j(k)| \leq K_{max,j}$, where $v_{max,j}$ is the maximal forward velocity of the j-th vehicle, $v_{min,j}$ is the limit on the backward velocity and $K_{max,j}$ is the maximal control curvature.

In the proposed method, we use the receding horizon control, that solves a finite horizon optimization control problem starting from current state $\psi(t_0)$ over the time interval $\langle t_0, t_0 + N\Delta t\rangle$. The duration $N\Delta t$ of the time interval $\langle t_0, t_0 + N\Delta t\rangle$ is known as the control horizon and N is number of transition points in the control horizon. After a solution from the optimization problem is obtained on a control horizon, a portion of the computed control actions is applied on the interval $\langle t_0, t_0 + n\Delta t\rangle$, known as the receding step. Parameter n is the number of transition points applied in one receding step. This process is then repeated on the interval $\langle t_0 + n\Delta t, t_0 + N\Delta t + n\Delta t\rangle$ as the finite horizon moves by *time steps* $n\Delta t$, yielding a state feedback control scheme strategy. See [4] for further details on RHC.

For the formation driving, we utilized a method in which the followers are maintained in relative distance to the virtual leader in curvilinear coordinates with two axes p and q, where p traces $\Psi_L(t)$ and q is perpendicular to p as is demonstrated in Fig. 2. The positive direction of p is defined from R_L back to the origin of the movement R_L and the positive direction of q is defined in the left half plane from the robots perspective.

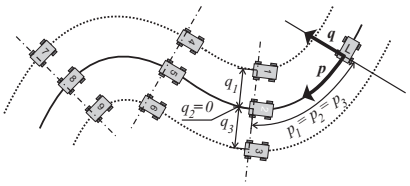


Fig. 2. Formation described by curvilinear coordinates p and q.

The shape of the formation is then uniquely determined by states $\psi_L(t_{p_i(t)}) = \{x_L(t_{p_i(t)}), y_L(t_{p_i(t)}), \theta_L(t_{p_i(t)})\}$, $i \in \{1, \ldots, n_r\}$, in *travelled distance* $p_i(t)$ from R_L along the virtual leader's trajectory and by *offset distance* $q_i(t_{p_i(t)})$ between $\bar{p}_L(t_{p_i(t)})$ and $\bar{p}_i(t)$ in perpendicular direction from the virtual leader's trajectory. $t_{p_i(t)}$ is the time when the virtual leader was at the *travelled distance* $p_i(t)$ behind the actual position. The parameters $p_i(t)$ and $q_i(t)$ can vary for each follower during the mission.

The following equations can be applied to convert the state of the followers in curvilinear coordinates to the state in rectangular coordinates:

$$\begin{aligned} x_i(t) &= x_L(t_{p_i(t)}) - q_i(t_{p_i(t)}) \sin(\theta_L(t_{p_i(t)})) \\ y_i(t) &= y_L(t_{p_i(t)}) + q_i(t_{p_i(t)}) \cos(\theta_L(t_{p_i(t)})) \\ \theta_i(t) &= \theta_L(t_{p_i(t)}). \end{aligned} \tag{2}$$

IV. Method description

A. Leader trajectory planning and control

In the application of airport snow shoveling, the formation should follow axes of the runways to cover the surface that is mainly used by the airplanes. The desired path for the virtual leader is therefore a sequence of connected axes of roads provided by the *Task Allocation* module, which was introduced in Fig. 1. Such a path composed from line segments is not feasible for the formation of car-like robots, but the proposed RHC based method can overcome the unsmooth connections.

Let us describe the k-th line segment of the desired path by equation $\varphi(k, s) = (P_k - P_{k-1})\, s + P_{k-1}$, where parameter s is within the interval $\langle 0, 1\rangle$. The points P_k, where $k \in \{1, \ldots, \tilde{n} - 1\}$, are crossings of axes of neighboring roads, P_0 is the beginning of the first axis and $P_{\tilde{n}}$ is the end of the last axis. The whole string of the segments is expressed as $\varphi(k, \cdot) = \{\varphi_x(k, \cdot), \varphi_y(k, \cdot)\}$, where $k \in \{1, \ldots, \tilde{n}\}$. Parameter $\tilde{n}$ is the number of roads provided by the *Task Allocation* module.

Having defined the desired path for the virtual leader, we can purpose the leader's trajectory planning and control approach appropriate for the purpose of airport snow shoveling. The aim of the method is to find a control sequence which could control the virtual leader along the runways axes by minimizing a given cost function. By applying this concept, the group should be able to respond to changes in workspace that can be dynamic or newly detected static obstacles. To define the trajectory planning problem in a compact form we need to gather states $\psi_L(k)$, where $k \in \{1, \ldots, N\}$, into vector $\Psi_{L,N} \in \mathbb{R}^{3N}$ and the control inputs $\bar{u}_j(k)$, where $k \in \{1, \ldots, N\}$, into vector $\mathcal{U}_{L,N} \in \mathbb{R}^{2N}$. All variables describing the trajectory of the virtual leader can be collected in an optimization vector, $\Omega_L = [\Psi_{L,N}, \mathcal{U}_{L,N}] \in \mathbb{R}^{5N}$.

The trajectory planning can be then transformed to the minimization of cost function $J_L(\Omega_L)$ subject to sets of equality constraints $h_{T_N}(\cdot)$ and inequality constraints $g_{T_N}(\cdot)$, $g_{r_{a,L}}(\cdot)$, that is

$$\begin{aligned} \min J_L(\Omega_L), \text{s.t. } & h_{T_N}(k) = 0, \forall k \in \{0, \ldots, N-1\} \\ & g_{T_N}(k) \le 0, \forall k \in \{1, \ldots, N\} \\ & g_{r_{a,L}}(\Omega_L, \mathcal{O}_{obs}) \le 0. \end{aligned} \tag{3}$$

The cost function $J_L(\Omega_L)$ is presented as

$$\begin{aligned} J_L(\Omega_L) &= \sum_{k=1}^{N} d\left(\varphi(\cdot,\cdot), \bar{p}_L(k)\right)^2 \\ &+ \alpha \sum_{j=1}^{n_0} \left(\min\left\{0, \frac{dist_j(\Omega_L, \mathcal{O}_{obs}) - r_s}{dist_j(\Omega_L, \mathcal{O}_{obs}) - r_a}\right\}\right)^2 \\ &+ \beta \left(\int_{ind_s}^{1} \sqrt{(\varphi'_x(ind_k, s))^2 + (\varphi'_y(ind_k, s))^2}\mathrm{d}s \right. \\ &\left. + \sum_{k=ind_k+1}^{\tilde{n}} \int_{0}^{1} \sqrt{(\varphi'_x(k, s))^2 + (\varphi'_y(k, s))^2}\mathrm{d}s \right)^{-1}, \end{aligned}$$

where the first term penalizes solutions with states deviated from the desired path. The influence of the environment on the final solution is added to the cost function in the second term. Function $d\left(\varphi(\cdot,\cdot), \bar{p}_L(k)\right)$ provides the minimal distance between the desired path $\varphi(\cdot,\cdot)$ and the position $\bar{p}_L(k)$. Function $dist_j(\Omega_L, \mathcal{O}_{obs})$ provides Euclidean distance between obstacle j and the virtual leader's trajectory. The third term of the objective function is important for the convergence of the method. This part is inversely proportional to the length of the path $\varphi(\cdot,\cdot)$ between the closest point on $\varphi(\cdot,\cdot)$ to the last state $\psi_L(N)$ and the desired end of $\varphi(\cdot,\cdot)$. It "pulls" via the constraints $h_{T_N}(\cdot)$ all states $\psi_L(k)$, $k \in \{1, \ldots, N\}$, along $\varphi(\cdot,\cdot)$ to the end of $\varphi(\cdot,\cdot)$. The variables ind_k and ind_s are indexing the closest point on $\varphi(\cdot,\cdot)$. Finally, the constants α and β are utilized for the balancing of frequently antagonistic endeavors: i) closely follow the desired path, ii) avoid dynamic obstacles and iii) reach the desired goal as soon as possible.

The kinematic model (1) with initial conditions given by the actual state of the virtual leader is represented using equality constraints $h_{T_N}(k)$, $\forall k \in \{0, \ldots, N-1\}$. This satisfies that the obtained trajectory stays feasible with respect to kinematics of nonholonomic robots. Set of constraints $g_{T_N}(k)$, $\forall k \in \{1, \ldots, N\}$ characterizes bounds on the velocity and curvature of the virtual leaders defined in III. Finally, the avoidance inequality constraints $g_{r_{a,L}}(\Omega_L, \mathcal{O}_{obs})$, which characterize the safety avoidance regions, are defined as $g_{r_{a,L}}(\Omega_L, \mathcal{O}_{obs}) := r_{a,L}^2 - dist_j(\Omega_L, \mathcal{O}_{obs})^2, j \in \{1, \ldots, n_0\}$.

B. Trajectory tracking for followers

The trajectory computed as the result of the previous section will be used as an input of the trajectory tracking for followers. First of all, such a solution needs to be transformed for each following vehicle via the equations (2). The obtained sequence $\psi_{d,i}(k) = (\bar{p}_{d,i}(k), \theta_{d,i}(k))$, where $k \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, n_r\}$, represents desired states for trajectory tracking algorithm with obstacle avoidance functions. Such a scheme enables to respond to events in environment behind the actual position of the virtual leader and to incorrect driving direction or velocity of the neighbors in the formation. In [8], one can find implementation details and experiments describing the abilities of dynamic obstacle avoidance as well as ploughs' failures tolerance, which cannot be presented here due to the space limitation.

C. Splitting and merging

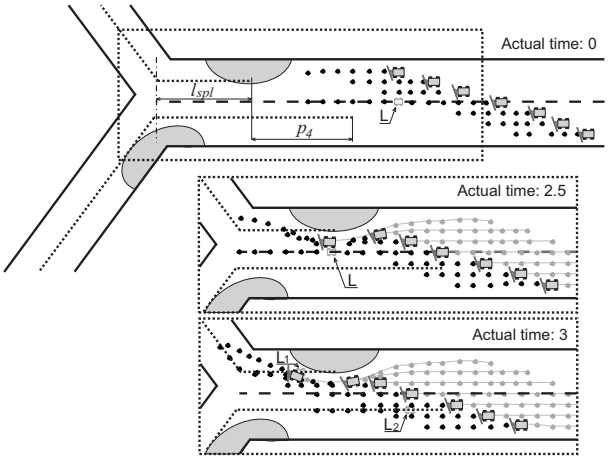
The approaches for splitting and merging of formations of snow-ploughs presented in this paper is crucial for the airport snow cleaning project. Both methods will be explained using specific examples containing all problems that need to be solved during the snow shoveling of airports.

The simulations presented in this section have been obtained using the proposed formation driving algorithm with settings: $N = 6$, $n = 2$, $\alpha = \beta = 1$ and $\Delta t = 0.25$s. Therefore the time difference between two subsequent planning steps in the simulations of splitting and merging is 0.5s.

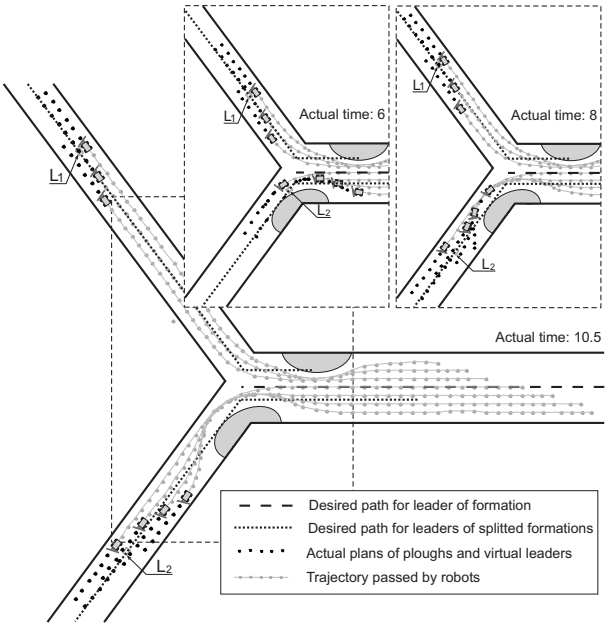
1) Splitting: The key problem of the splitting approach is to find an appropriate time when to transfer the leading tasks from a single virtual leader to several virtual leaders that belong to the lately formed formations. The splitting point should be postponed to as late as possible as arises from the formation driving principles. The ploughs connected to a team could better avoid collisions within the formation and with obstacles and also the complexity of task allocation and communication is decreased.

Let us now analyze the problem from the opposite view. The desired path that is followed by the virtual leader of the big formation will differ from those used by the new virtual leaders as you can see from Fig. 3. In addition, we can suppose, that the new formations have to change their heading for cleaning the following road during the splitting. To take advantage of the whole control horizon, the new desired path should be utilized when the new direction of movement can influence the optimization process. For simplification we can say, that the formation should be divided under the commands of new virtual leaders in the distance of the former virtual leader from the center of the crossroad greater than or equal to the length of the planning horizon. Due to the prior knowledge of the maximal leader's speed we can consider an upper bound l_{spl} of the length of the planning horizon as $l_{spl} = N\Delta t \max_{\tau \in \langle t; t+N\Delta t \rangle}(v_{max,L}(\tau))$.

We can conclude that the formation will be splitted in the distance from the center of the crossroad equal to l_{spl} to satisfy both requirements mentioned above: 1) keep the robots in the big formation as long as possible and 2) switch a



(a) $t = 0$s - big formation cleaning the main runway; $t = 2.5s$ - formation closely before the splitting; $t = 3$s - formations that have just been split



(b) $t = 6$s - formation led by L_2 is avoiding an obstacle partly blocking the cleaned road; $t = 8$s - formation led by L_2 is going back to the desired shape; $t = 10.5$s - complete history of the formations' movement

Fig. 3. Formations' movement during simultaneous splitting and obstacle avoidance.

desired path sufficiently far from the crossroad. Perceive that both, the path for the big formation as well as the new desired paths, must be overlapping to employ the RHC concept. Before the switch-point, the previous path contributes to the cost function of the virtual leader. This virtual point is situated on the axis of the formation and in the way that $p_1(\cdot)$ coordinate of the first plough in the formation is zero. Once the formation reaches the switch-point, the new virtual leaders are placed on the axes of established formations again next to the first ploughs. Therefore, we have to specify the minimum overlapping of the desired path for leaders of subformations and the desired path for the leader of the splitted formation as $l_{spl} + p_\epsilon$, where p_ϵ is coordinate of the first

robot of the sub-formation in the big formation. An example of a formation splitting is presented in Fig. 3.

2) Merging: The core idea of the approach for merging of several sub-formations is to iteratively divide the merging process to the simple jointing of two formations as illustrated in the following simulation. In Fig. 4 ($t = 0$s), the formations F_1, F_2 and F_3 guided by the virtual leaders L_1, L_2 and L_3 are approaching to the crossroad. The formations should pass through the crossroad in the order in which they will be maintained in the big formation. This procedure should minimize snow remaining on the runway as well as decrease the collision risk. Due to the shovels' orientation, the ploughs must be arranged in sequence F_1, F_2, F_3 from the front to the back of the composed shoveling column.

The snapshots presented in Fig. 4 could help to clarify the merging process. In the snapshots the formations are appropriately adapting their velocities to be on time in the merging position. It could be difficult to estimate the optimal beginning of the acceleration to ensure fluent coupling in a real experiment. Fortunately, the inaccuracy in the spacing between the sub-formations can be suppressed using the periodical replanning during the movement.

Similarly to the formation splitting, the formation merging is also restricted by two antagonistic requirements: 1) the formations should be merged as soon as possible, 2) the virtual leaders have to follow parallel paths at time of merging. Therefore, the formations are always merged when the position of the virtual leader of the second formation is behind the crossroad of their runways.[1] The paths that should be followed by the virtual leaders during the formation merging are again overlapping because of the employed RHC approach. The following simple rules can be iteratively utilized for the paths' construction: i) $l_{spl} + p_\epsilon$ is distance between the point of merging and the end of the path followed by the first sub-formation. ii) l_{spl} is distance between the point of merging and the end of the path followed by the second sub-formation. iii) p_μ is distance between the point of merging and the beginning of the path followed by the merged formation.

p_μ is coordinate of the first robot of the second sub-formation in the big formation. Let us describe the example in Fig. 4 to clarify these rules. In the first step of iterative merging, the formation F_2 is attached to the formation F_1. The path for F_1 is extended according i) with length $l_{spl}+p_4$ behind the crossroad, because the first plough of F_2 is going to be the fourth in merged formation F_{12}. The path for F_2 is extended according ii) with length l_{spl}. The path which will be followed by F_{12} begins in distance p_4 from the crossroad. In the next step of iterative merging, the formation F_3 is attached to the formation F_{12} and the first plough of F_3 becomes the sixth in the final formation F_{123}. Therefore, the path for F_{12} finishes in distance $l_{spl}+p_6$ from the crossroad and the path for F_3 in distance l_{spl}. The path for F_{123} begins in distance p_6 from the crossroad.

[1]In the snapshot in $t = 0$s this point matches the crossing of the dash desired path and the perpendicular dot-dash line.

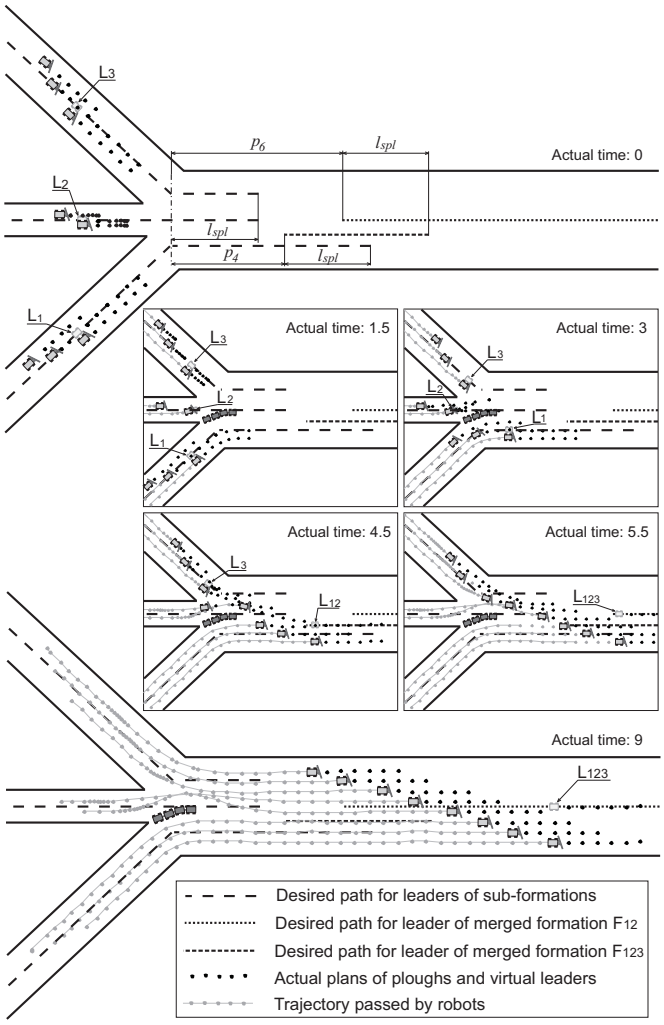


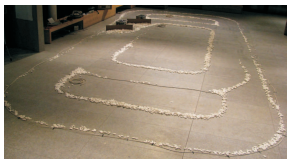
Fig. 4. Formations' merging maneuver: $t = 0$s - three small formations shoveling auxiliary roads; $t = 1.5$s - formations F_2 and F_3 modify their velocities with a goal to continuous connection. Formation F_2 is avoiding the detected unforseen obstacle; $t = 3$s - formation F_2 accelerates with an aim to join formation F_1. Formation F_3 waits for the formations going by; $t = 4.5$s - formations F_1 and F_2 are already merged to formation F_{12} and formation F_3 accelerates to join them; $t = 5.5$s - all formations are just merged to one big formation $F_{1,2,3}$; $t = 9$s - the complete history of the formations' movement during the merging.

V. HARDWARE EXPERIMENT

This section describes the snow shoveling hardware experiments that were carried out with the G2Bot-Testbed[2] of the Czech Technical University. The G2Bot robotic platform is equipped with sensors for odometry and wireless communication, which has been used for distribution of data necessary for the formation stabilization. Control inputs have been computed on-line on robot's internal PC. For the experiments the snow was made of small pieces of polystyrene. We used straight bars mounted on the robots as shovels. The experimental scenario consists of two larger runways that has to be swept by two vehicles and four smaller roads for only one plough. Initially, the formation parameters are

[2]The G2Bot is a differential drive robot with a function emulating car-like robots kinematics.

(a) Initial position of ploughs.



(b) The ploughs have completed one cleaning cycle.

(c) Cooperative shoveling of the runway.

(d) Splitting to two independent units.

(e) Re-building of the formation.

(f) Shoveling of the 2nd runway.

(g) The formation is again splitted.

(h) Beginning of the second splitting.

Fig. 5. Snapshots from snow shoveling hardware experiment.

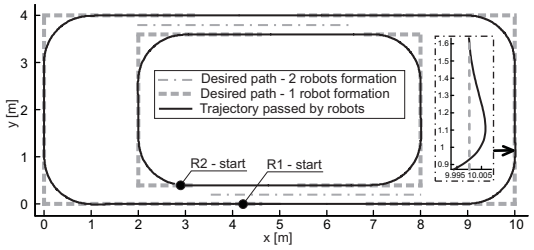


Fig. 6. Real data captured from the odometry of the ploughs.

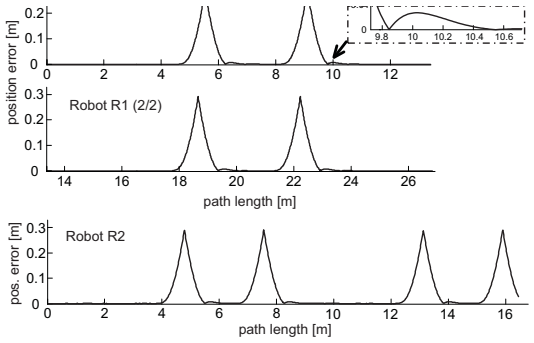


Fig. 7. Position error between desired paths of robots and real paths passed by ploughs during the shoveling experiment.

chosen to be $p1 = 0$m, $p2 = -1.25$m, $q1 = 0.2$m and $q2 = -0.2$m. The maximal speed of ploughs has been limited to 0.07m/s due to the utilized simple shovels and light polystyrene imitating the snow. The formation driving algorithm introduced in Section IV-A has been used with settings $N = 14$, $n = 2$, $\alpha = \beta = 1$ and $\Delta t = 2$s. Therefore the length of the control horizon is 1.96m for ploughs going with maximal speed. This enables to efficiently cover the surface of runways in sharp corners of the desired path. Time difference between two subsequent planning steps is 4s.[3]

Fig. 5 shows snapshots from one of the experimental runs and data obtained from the odometry of ploughs has been plotted in Fig. 6. The desired paths for virtual leaders of formations, which have been designed using the guideline from Section IV-C, are also depicted in Fig. 6. Errors between the position of the robots, which is obtained from the odometry, and the desired position, which is computed using (2), are plotted in Fig. 7. The high peaks in the graph corresponds with parts of the path around the corner of the workspace.[4] The small overshoots behind these peaks (see the zoomed part in Fig. 7) are much more interesting for the system analysis. It shows, that the error of the controller is less than 1cm.

VI. CONCLUSIONS

In this paper, we described an approach for formation driving of autonomous ploughs in the task of airport snow shoveling. Our method utilizes the RHC for the purpose of the optimal coverage in environment with static and dynamic obstacles. We extended the approach with abilities to form temporary formations, to split formations to several smaller teams and to merge the sub-formations. The developed methods as well as the complete system were verified by various simulations and hardware experiments.

[3]Maximal computational time needed for the leader or followers planning has been 3.23s using *fmincon* solver in MATLAB environment on the internal 1.2 GHz PC with 1GB RAM. Therefore the plans could be computed on-line during the movement of robots.

[4]These peaks are not a fault of the system but its characteristics. The fluent turning is preferable to the sharp edge in the simple desired path.

REFERENCES

[1] L. Consolini, F. Morbidi, D. Prattichizzo, and M. Tosques, "Leader-follower formation control of nonholonomic mobile robots with input constraints," *Automatica*, vol. 44, no. 5, pp. 1343–1349, 2008.

[2] R. Fierro, A. Das, V. Kumar, and J. Ostrowski, "Hybrid control of formations of robots," in *Proc. of IEEE Conference on Robotics and Automation*, 2001.

[3] M. Hess, M. Saska, and K. Schilling, "Application of coordinated multi vehicle formations for snow shoveling on airports," *Inteligent Service Robotics*, vol. 2, no. 4, pp. 205 – 217, October 2009.

[4] T. Keviczky, F. Borrelli, K. Fregene, D. Godbole, and G. Balas, "Decentralized receding horizon control and coordination of autonomous vehicle formations," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 1, pp. 19–33, 2008.

[5] D. Kurabayashi, J. Ota, T. Arai, and E. Yoshida, "Cooperative sweeping by multiple mobile robots," in *Proc. of the IEEE International Conference on Robotics and Automation*, 1996.

[6] R. Kurazume and S. Hirose, "Development of a cleaning robot system with cooperative positioning system," *Autonomous Robots*, vol. 9, no. 3, pp. 237 – 246, 2000.

[7] M. Saska, M. Hess, and K. Schilling, "Route scheduling approach for airport snow shoveling using formations of autonomous ploughs," in *Proc. of 10th International Conference on Control, Automation, Robotics and Vision*, 2008.

[8] M. Saska, J. S. Mejia, D. M. Stipanovic, and K. Schilling, "Control and navigation of formations of car-like robots on a receding horizon," in *Proc of 3rd IEEE Multi-conference on Systems and Control*, 2009.

[9] H. Tanner, G. Pappas, and V. Kumar, "Leader-to-formation stability," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 443–455, June 2004.

[10] I. A. Wagner, Y. Altshuler, V. Yanovski, and A. M. Bruckstein, "Cooperative cleaners: A study in ant robotics," *The International Journal of Robotics Research*, vol. 27, no. 1, pp. 127–151, 2008.