INTELLIGENT AUTOMATION & SOFT COMPUTING, 2017
http://dx.doi.org/10.1080/10798587.2016.1264695

Mobile Robots Navigation Modeling in Known 2D Environment Based on Petri Nets

S. Bartkevicius[a], O. Fiodorova[b], A. Knys[c], A. Derviniene[d], G. Dervinis[c], V. Raudonis[c], A. Lipnickas[c], V. Baranauskas[c], K. Sarkauskas[c] and L. Balasevicius[c]

[a]Department of Electrical Power Systems, Kaunas University of Technology, Kaunas, Lithuania; [b]Department of Automation, Kaunas University of Technology, Panevezys, Lithuania; [c]Department of Automation, Kaunas University of Technology, Kaunas, Lithuania; [d]Department of Electronics Engineering, Kaunas University of Technology, Kaunas, Lithuania

ABSTRACT

The paper deals with supervised robot navigation in known environments. The navigation task is divided into two parts, where one part of the navigation is done by the supervisor system i.e. the system sets the vector marks on the salient edges of the virtual environment map and guides the robot to reach these marks. Mobile robots have to perform a specific task according to the given paths and solve the local obstacles avoidance individually. The salient point's detection, vector mark estimation and optimal path calculation are done on the supervisor computer using colored Petri nets. The proposed approach was extended to simulate a flexible manufacturing system consisting of swarm of 17 robots, 17 - warehouses and 17 - manufacturing places. Our experimental investigation showed that simulated mobile robots with proposed supervision system were efficiently moving on the planned path.

KEYWORDS

Mobile robots; Robot navigation; Vectors tree; Petri nets

1. Introduction

Perhaps the main purpose of the mobile autonomous robots is their usage in an industrial manufacturing or warehouse environment. Today most of mobile robots, such as automatic dust cleaners or lawn mowing robots, execute certain tasks in a strictly known environment. Usually mobile robots consist of two parts; hardware and software. The hardware part is responsible for wheels or tools control and for collecting information from external sensors. The software part is responsible for input signal processing and decision-making, i.e. to localize the robot in the environment and to control its motions based on external and internal sensors information (Azarm & Schmidt, 1997; Satish & Madhava, 2007; Yi, Tao, & Xiaqin, 2012a). For example, there are a number of existing cordless automatic lawn mowers in the market (Borenstein, Everett, Feng, & Wehe, 1999; Burguera, Gonz′alez, & Oliver, 2009; Defoort, Kokosy, Floquet, Perruquetti, & Palos, 2009; Sooyong & Jae-Bok, 2005) with some advantages (autonomous multi zone cutting capability) and disadvantages (battery or solar power).

Therefore, when mobile robots are used as transportation units in a manufacture process or in the warehouse, the external supervising control system is needed to control the individual robot or swarm of robots in order to ensure the execution sequence of given tasks. The most common way is to design a robotic system in hierarchical structure, where relatively simple tasks such as detection of an object, grasping or releasing an object are done by the mobile robot itself. All computational tasks dealing with navigation, motion trajectory planning (Omar & Yahya, 2007; Weiren, Kai, & Simon, 2009), and prediction of battery charging time must be executed on a supervisory system. The robot itself should be simple and execute only local tasks (obstacle avoidance, energy saving, etc.); the supervisory system should execute global tasks.

The CAM (Computer aided manufacturing) system is a good example of a hierarchical control system, which controls robots in certain known environments. The robots are fixed to the coordinates of the working area or they have limited reaching space, certain parts from one working area to another are transported (like conveyor applications). However, the exclusion of CAM conveyor system automatically raises the question about the use of mobile robots for such system service. The CAM system plans manufacturing tasks, distributes them to the appropriate machine tools, and uses mobile robot or swarms of them to solve the task. CAM system controls machines, their preventive and scheduled maintenance, when the same manufacturing components road changes in time. So it is impossible to anticipate all the possible work situations. This system requires larger capacity mobile robots with a powerful control system for the service tasks.

Significantly smaller robots can be used in flexible manufacturing systems for item transportation purposes. In this case, the robot control system is simpler, because they only need to be able to send and receive navigation commands and to take the necessary item and transport it to the specified location (Berg, Patil, Sewall, Manocha, & Lin, 2008; Khashayar & Arvin, 2003; Kluge & Prassler, 2004). Main supervisory system task in the known environment, as in the CAM system, is to set movement trajectories and transfer this information to mobile robots, or to organize the known environment in new territory and carry out the same tasks as the CAM system.

The authors of the work use so called vector marks to evaluate the best/shortest motion trajectory for mobile robots. The motion of the mobile robots is controlled using the supervision system, which autonomously tracks each robot independently and estimates the optimal path. Experimental results are obtained using colored Petri nets.

CONTACT V. Baranauskas virginijus.baranauskas@ktu.lt

The paper is organized into four sections. First section introduces the reader with the problem. Second chapter gives more detailed information about proposed path planning algorithm using colored Petri nets. Experimental investigations and results are discussed in the Third section. Possibilities of swarm of robots usage and control of it in flexible manufacturing system are presented in Fourth section. Conclusions and references are presented at the end of the paper.

2. Optimal Path Planning Approach

The optimal motion trajectory of the mobile robot is estimated using vector marks, which are generated from the scanned contours of the surrounding environment. The aim of scanning pre-processing is to detect any alterations, jumps or cracks in a contour of scanned obstacles. The most informative abnormalities are further named as salient points. During the scanned data analysis all abnormalities should be identified and all sudden gradient direction changes should be marked. In such an approach it is enough to determine the tangential angles in a point of interest instead of calculating the exact gradient at each point. The presented research in this article is based on such an assumption and is aimed to detect the salient points rather than gradient alteration at all points (Baranauskas, Derviniene, Sarkauskas, & Bartkevicius, 2010). Scanning system (lidar) scans the surroundings with fixed angle step from the target point and detects all possible obstacles around it. The proposed methods introduce a new concept—an abrupt scan data point, which can be later named as simple "jump" or "salient" point.

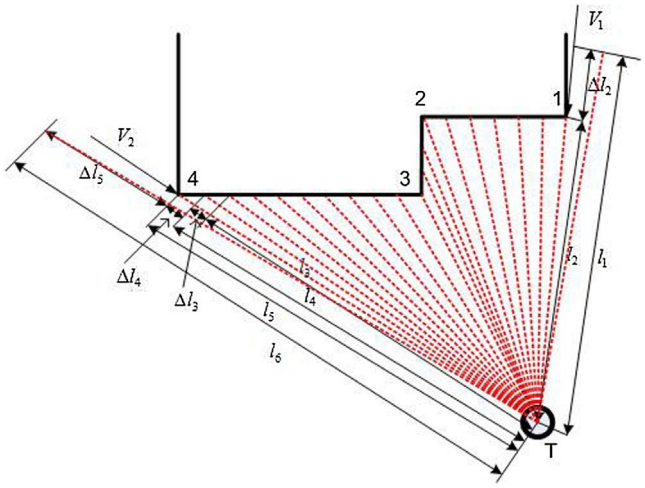

Figure 1. Detection of Alternating Points (1,4—are Salient and 2, 3—are Jump Points).

An abrupt data point is a point, which is detected when scanning the surroundings with obstacles and it coincides with a jump point of the edge of a static obstacle. However, not all jump points of the obstacles can be considered as important or salient points. An important salient point has a specific property; beside this point in the scanned surrounding should be an emptiness at a length of the scanner radar beam prior the next scanned data point. All other alternations of scanned data are called as jumps, because they are shorter than maximum scanner range distance. This situation is demonstrated in Figure 1.

In Figure 1 the static obstacle with four corners is scanned in the known environment with a fixed scanning angle. From the target point T the scanner scans the surroundings within an established scan angle by the step $\Delta\alpha$. The scan starts from right to left; then the first distance to the obstacle is found as l_1.In the next scanner step the distance to the obstacle is l_2 and the step difference at corner (1) is Δl_2. Therefore at corner (1) a virtual vector, oriented along the nearest fragment of a path with head in vertex (salient point) can be assigned and called as *vector* mark. In other words, this vector consists of a set of data pointing to direction along the path towards the target, which is possible in the turning point of a path. These marks are direction vectors or crossroad signs guiding the mobile robot to move towards the final destination. Only the first marks can be considered as the vectors of the direct course since they are directly pointed at the target point. All other vector marks in respect of static obstacles show the direction following, which the target point can be reached in the shortest path. These vector marks for a robot gives information as traffic direction signs on roads for humans. From Figure 1 it is obvious to see that the salient points of this figure are corners 1 and 4, where the vector marks V_1 and V_2 are placed. The corners 2 and 3 are out of interest, because they are not guiding how to drive the robot to the target position by the shortest path.

Figure 2 demonstrates three situations in which the length of the vector mark A and further size of the scanning angle α (wide angle) and β (blind angle) are different. In Figure 2a, the scanning angle α is 270°, blind-angle β is 90°. In this case, all possible salient points can be found. Figure 2c presents the situation with the scanning angle α is reduced to 180°; as it is seen the hole in the obstacle is not detected as well as corner *"5"*is unseen. If the vector mark A is too long, as it is shown in Figure 2b, and the scanning angle α is 270°, the hole inside the obstacle could also be unseen. In this case, it is necessary to increase the size of the scanning angle α. Theoretically, it is hard to establish exact range of the size of the scanning angle α; however, the relation between the length of a vector mark and the size of the scanning angle is obvious. It is possible to

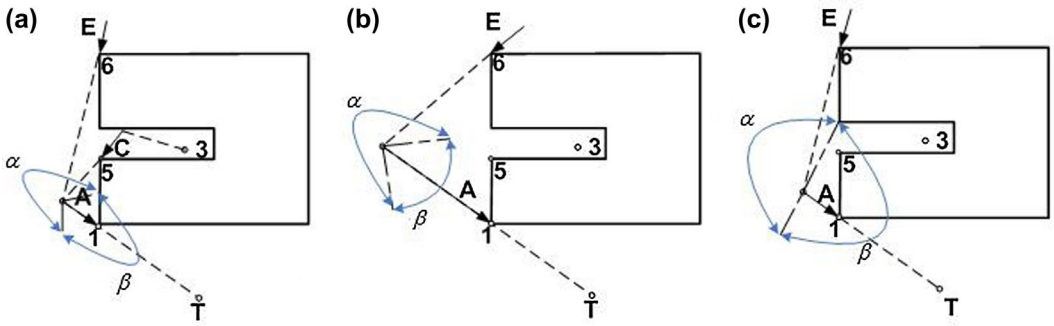

Figure 2. Dependence of the Detection of a Salient Point and Formation of Vector Marks upon the Length of an Older Generation Vector Mark and the Size of the Scanning Angle.

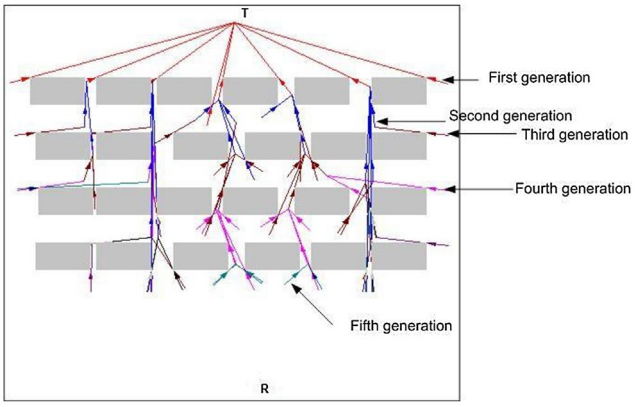

Figure 3. Selected Situations Vector Tree, Generated from the Target Point T in Known Environment.

conclude that the longer a vector mark is the larger scanning angle α has to be used also. Each vector mark has its own characteristic—weight coefficient. Weight coefficient of a vector mark is a sum distance from the end point of a vector mark to the target point. In other words, the weight coefficient is the distance from the target point to the end of a vector mark.

The first vector marks formed from the target point are of the oldest generation and it goes toward to robot position. The weight coefficient of younger generation vector marks will equal the sum of the weight coefficients of older generation vector marks and the distance to the younger generation vector mark, i.e.

$$D_1 = D_{v1} + dd_1, \tag{1}$$

$$D_2 = D_1 + D_{v2} + dd_2, \tag{2}$$

$$D_3 = D_2 + D_{v3} + dd_3, \tag{3}$$

$$D_i = D_1 + \sum_{l=2}^{i} \left(D_{vn} + dd_n\right), i = 2, 3, ...; n = 2, 3, ...; \tag{4}$$

Here D_1 is the weight coefficient of the first vector mark, D_{v1} is the distance from the vertex point of the first vector mark to the target point, dd_1 is the length of the first vector mark, D_{v2} is the distance from the end point of the first generation vector mark to the vertex point of the second generation vector mark, dd_2 is the length of the second vector mark, D_{vn} is the length of the vector mark n from its vertex to the end point of the older generation vector mark, dd_n is the length of the vector mark n, D_i is the weight coefficient (the distance to the target point) of the vector mark i.

Distances between entire position point and visible vector mark must be evaluated calculating distance to the target point. This distance can be significant and important. If you calculate distance from the point "O_R" using different vector marks from right or left side in Figure 2, total distance to the target point may be not the same. It depends on which vector mark is selected. Authors use criterionλ, in order to select vector, which can lead to the target point in the shortest path, from visible vectors k:

$$\lambda = \min_{n=1}^{k} \left(W_n + \left\|p_{nv} - p_r\right\| + \left\|p_{nh} - p_{nv}\right\|\right) n \in \mathfrak{R}, \tag{5}$$

Where W_n—weight coefficient of the n-th vector mark, p_{nv}—visible vector mark point, p_r—robot position, p_{nh}—apex of the vector mark.

All parameters, describing the vectors, are tied to an index vector end, i.e. to the corner from which the vector is generated. The vector has the length of the specified size, in order to be able to easily find other corners of the current environment situation, such as the holes in the wall or door, since the next vectors are generated from a non-end vector direction (Baranauskas V. et al., 2010). In this way the holes and holes in parallel planes are found. All this is illustrated in Figure 3. There we see different generations of formed vector marks in known scanned environment.

The experimental testing of vector mark tree formation (i.e. number of formed vector marks depending on the length of vector and scanning angle α) is made in the virtual environment, which was showed in Figure 3, are shown in Figure 4 and Figure 5. The place of target point is the same. From Figure 4 we see, that the number of generated vector marks decreases significantly, when the length of the vector mark is more than 5. There is not big difference on the number of generated vector marks, when the length of it is from 1 to 3. We can notice that the number of generated vector marks is almost stable, when the length of vector mark varies from 10 to 20. In this case, the length of formed vector mark is shortened, in order to find all break points and fit among the obstacles.

From Figure 5 we see, that the number of generated vector mark significantly increases, when the scanning angle varies from 20° to 180°. There is not big difference on the number of generated vector marks, when the scanning angle varies from 270° to 320°. So, for further investigations we use 320° scanning angle and length of vector mark is 3.

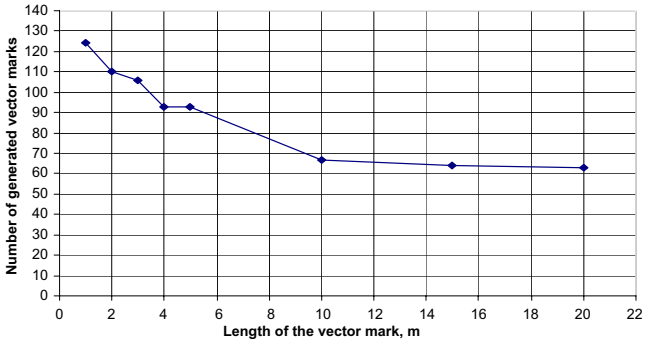

Figure 4. Dependence of Vector Marks Generation on the Length of the Mark, for the Environment, Shown in Figure 3.

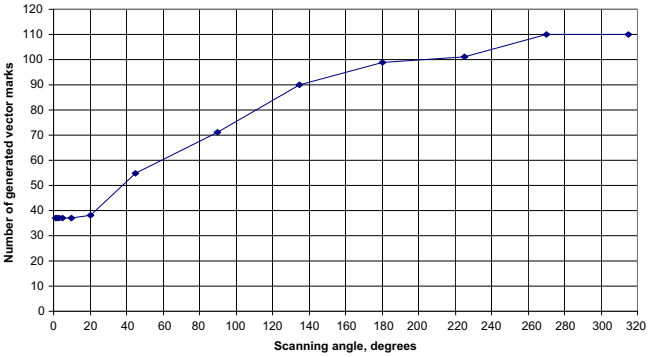

Figure 5. Dependence of Vector Marks Generation on the Scanning Angle, for the Environment, Shown in Figure 3.

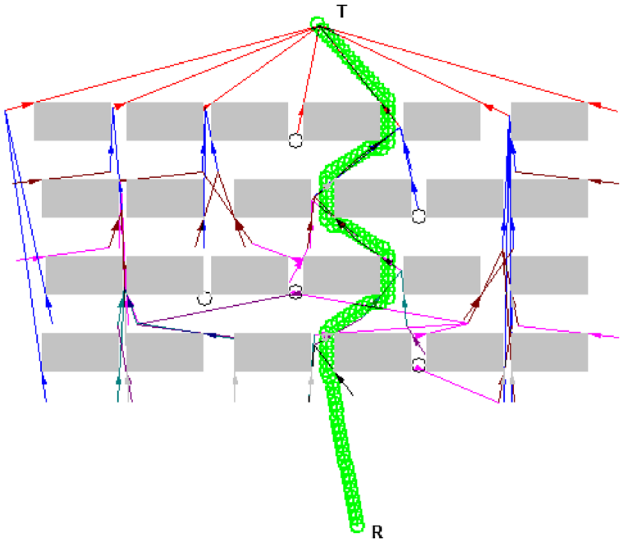

Figure 6. Particular Dimensions Robot can pass not through all the Openings, so Optimal Trajectory is Found from the Shortest Path, through which the Robot Can Pass. The Circles Indicate Places, where an Attempt was made in Order to find a way through places, because it is Shorter.

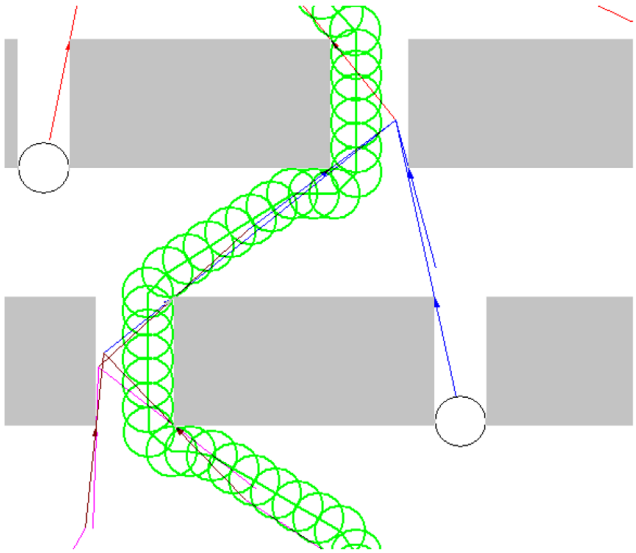

Figure 7. Mobile Robot Tracks Fragment Shows how the Robot Appears to Track the Assessed Dimensions. Two Holes are closed by Supervisor, because they are too narrow.

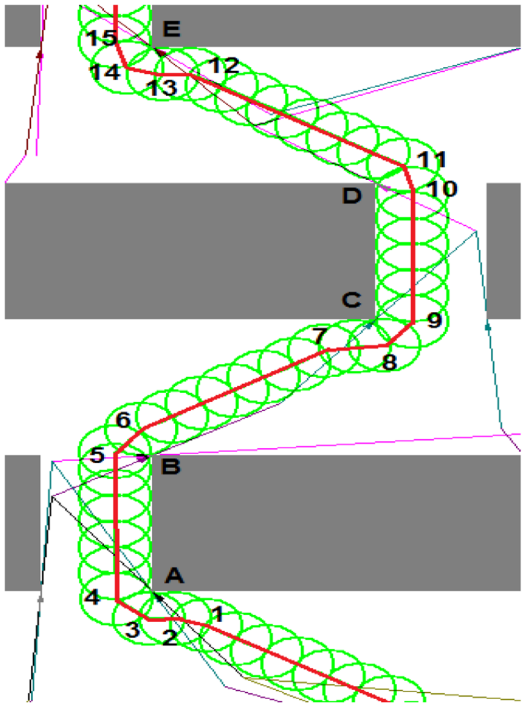

Figure 8. Movement of the Robot, According to the Supervisory System Generated Vector Marks Tree, according to the Given Path. Local Navigation Problems, such as Going around Corners of the Obstacles, the Robot Solves using Sensors Data.

Pseudo code 1: Estimation of optimal motion trajectory:

angle = 0; //orientation of the laser beam*SalientPoints[];* // the coordinates of salient points*RealSalientPoints*[]; *NmbOfPoint* = 0;Starting scanning with LIDAR range scanner**While** *Angle > 360* LIDAR scans environment trying to detect breaks of profile got Move the laser beam of range meter to the *Angle* and fix distance *Range*; *angle*++; add salient points to array *Salient-Points*;**END WHILE; FOR** n = 1 **TO** *SalientPoints.size()* **DO** The new branches of tree can be produced when salient points are found Add salient points to array *RealSalientPoints*[]; *NmbOfPoint*++;**END FORFOR** n := 1 **TO** *NmbOf-Point* **DOIF** new generated mark is not covered by an obstacle a mark of previously generated mark and it is not "father" **THEN** A new mark is redundant, if the distance to the target of new generated mark is larger of the same parameter of visible mark more than distance between both marks; Sign the mark as redundant; **ELSE IF** new generated mark "see" another mark of the same generation **THEN** A new mark is redundant, if the distance to the target of it and another mark is larger of the same parameter of visible mark more than distance between both marks; Clear sign of redundancy of the "seen" mark, if the sign exists; Sign current mark as redundant, but do not remove from *Tree* array;**END IF;END FOR;**Remove all signed as redundant marks from the array of vector marks;

However, mobile robots cannot cross all paths, because of its dimensions. Some of the holes are narrow or too small for a real robot. The supervisory system solves this situation by modelling the robot's trajectory and evaluating the robot dimensions, finding out which path is optimal passage and transmitting the coordinates of the path to the robot. The robot, in this case, uses the optimal path coordinates, because all situations that are not available are eliminated from the possible paths choice. All this is shown in Figure 6.

The robot must react to unexpected obstacles and use its scanners and sensors in order to eliminate deviations from the path, or get the correction information from supervisor system. The optimal motion trajectory is generated using algorithm shown below. A laser scanner LIDAR is used on each mobile robot in proposed situation (Baranauskas V. et al., 2003).

3. Application of Proposed Method to Obstacle Avoidance

Figure 7 shows a fragment of mobile robot paths. When the robot moves, it occupies a certain area and moves to vector tag hampers an obstacle. Such situations can appear, when the supervisory system generates a vector tree without the size information of mobile robots and the cavities in the obstacles. Then vector tree becomes universal and can be used for any mobile robot of different dimensions. Only the supervisory system determines through which holes of known environments may pass a robot of specific size. Now, the navigation task can be solved in two ways. The situation explained in Figure 8, which is a part of Figure 3 fragment.

Generated vectors coordinates are transmitted to mobile robot. The robot moves to vector A, but in its way touches the obstacle in point 1. Because of it, the mobile robot organizes

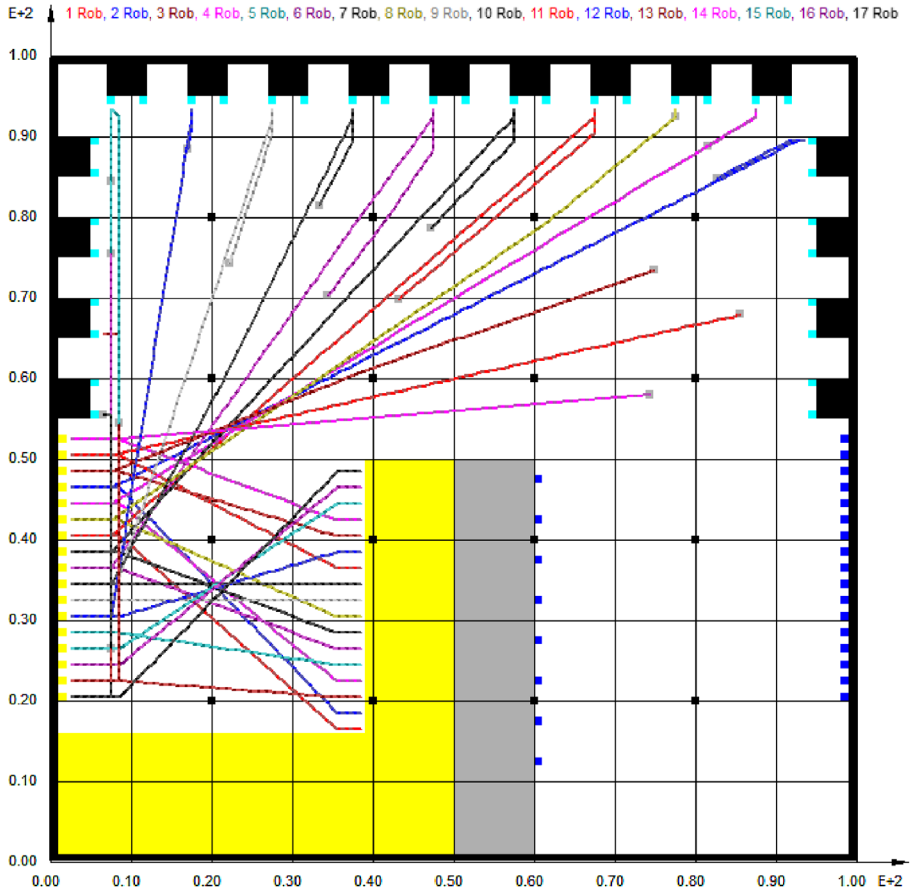

Figure 9. Experimental results.

bypass using middle points 2, 3 and 4 below. As mentioned before, in this case, there are two possible ways for solving this situation. First; the robot reaches point 1, in aid of its sensors performs bypass of the obstacle corner in order to reach the track further reference point. If talking about autonomous and Multi-Agent systems, when reaching the point 1 of obstacle circumvention direction depends on the programs in bypass mode; left, right, or random direction (Guo & Parker, 2002; Jur van den Berg, Ming, 2008; Yi, Tao, & Xiaqin, 2012b).

So a chosen direction will always be suitable in the sense that the distance to the final target point will always be shortest. Path of such navigation of the mobile robot is presented in Figure 8. Coordinates of points A, B, C, D and E are transmitted to the mobile robot. The robot starts moving toward the point A and reaches obstacle in point 1. Then the robot uses its sensory system and will move towards point B, depending on the obstacle, until points 2 and 3. The robot reaches point 4 on its way around the obstacle corner. Point B is in the obstacle corner, and the mobile robot moves along the obstacle using its sensory system. In this way, the supervisor to a mobile robot transmits only the characteristic points, which characterize the global direction of movement and the robot itself decides navigation details. How the robot solves different navigation problems around various forms of obstacles is described in detail in (Baranauskas V. et al., 2003).

Servicing the manufacturing process in a known environment, there are no such difficult situations environmental profiles and the realization of mobile robot navigation is significantly simpler. The situation looks different if mobile robots are used as vehicles and service depots. Figure 2, Figure 3, Figure 6 andFigure 8 show situations of mobile robot navigation, servicing of marine containers, when loading and unloading work goes on.

4. The Swarm Control of Mobile Robots

A swarm of small robots (with dimensions less than 100 cm) can be used in flexible manufacturing systems. Manufacturing tasks can successfully be executed by such robots, because their dimensions are acceptable in the manufacturing process for transportation. The usual task for them would be to move an item from one working area to other. The number of robots, used in manufacturing, depends on number of working positions. It must be guaranteed that the workflow must be continuous, i.e. without outage. Depending on the intensity of work, some robots can be out of working area, i.e. it could charge batteries. It means that the number of used robots must be large then 10 and if we want to optimize functionality of manufacturing process, it is necessary to have at least one robot free of work.

Movement conditions are defined by the supervisory system, which uses 3D cameras in order to determine the coordinates of the destination and the type of object robot has to take. Depending on the construction of the robot, it can take only a single object, or several identical objects, and then transport them.

Supervisory system observes the robot path and if it deviates from the set direction, due to possible road surface roughness, supervisor corrects its movement. System observes robot movements until the task is fully executed. If the robot has enough energy, it is sent to carry out the next task; otherwise it returns to the own cell in the hive for energy supply.

Experiments on swarm of robots control was made on the basis of the colored Petri nets. The model of manufacturing line was created using colored Petri nets in the programming package "Centaurus CPN". This software allows simulating workflow of the manufacturing line and it shows movement of robots at the same time. All experiments were executed using the programming package in a virtual environment.

Dimensions of the manufacturing area are 100x100 m and it is shown in Figure 9. This area is divided into several parts: Yellow area is reserved for people and it is robots garage points; grey area is reserved for robots charging places; white area is reserved for manufacturing process, where people can't enter. It is possible to use up to 17 robots in our experiments. There are 17 places for item picking, 17 manufacturing machines (each of it has 2 places; one is reserved for initial position, other is reserved for final product), 17 warehouses and 17 mobile robot charging places in the experimental area. There are building construction colons inside the area; it is showed as black small quadrates in Figure 9.

Figure 9 shows experimental results of robots swarm, which serves the manufacturing process control. Robots start movement from their locations in defined time intervals. It is necessary to set start delay for each robot, because otherwise robots will collapse with each other. Also, there are some places, where robots paths intersect.

CAM system is responsible for tasks distribution among manufacturing devices and robots, taking into account prophylaxis, scheduled repairmen. As manufacturing tasks are distributed among devices, it is impossible to set calculated paths of exact robot. Supervisory system, which is part of CAM, is responsible for task assigning to exact robot and calculations of paths. Supervisory system calculates robots path from its current position to primary work position and after that calculates robot movement path to final destination. Supervisory system knows start and target points of each robot and calculates its path according formed vector marks tree. Each robot must solve its own local navigation tasks, using its sensors. Also, it must execute main task - move along given path coordinates, observing coordinates of path turning points, movement direction, stop points and speed changing moments. There is only one possibility to optimize such swarm work, i. e. optimize a path of each robot in a swarm. Optimization criteria could be the length of the calculated path. When a robot fulfils the given task, it stays at the same point in area. If there are several free of work robots, supervisory system will select the nearest robot, to which it will assign the new task.

Figure 9 shows experimental results of robot swarm control. Using the current software, it is possible to set number of used robots up to 17. The paths of each robot are calculated by the supervisory system. Solving problems of possible robot collisions is out of scope of this paper.

5. Conclusion

This paper presents a supervised robot navigation in known environments. The global robot navigation is done by the supervisor system i.e. the systems sets the vector marks on the vertexes or salient points of the virtual environment map and guides the robot to reach these marks till the target point. The calculated vector consists of a set of data pointing to the direction along the path towards to the target and it is usually placed on the turning point of the path. However, the mobile robot cannot cross all paths, because of its real size. Some of the passages are narrow for a real robot. Supervisory system solves this situation by modelling the robot's trajectory by evaluating the robots dimension and finds out the shortest possible passage. The mobile robots have to perform a specific task according to the given paths and solve the local tasks individually.

The salient points detection, vector mark estimation and optimal path calculation is done on a supervisors computer by colored Petri nets. The proposed approach was extended to simulate a flexible manufacturing system consisting of swarm of 17 robots, 17 - warehouses and 17 - manufacturing places. Our experimental investigation has shown that simulated mobile robots with proposed supervision system were efficiently moving on the planned path.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Stanislovas Bartkevičius received a Ph.D. degree in Technical Cybernetics from Kaunas University of Technology, Lithuania, in 1975. He is currently a professor of the Faculty of Electrical and Electronics Engineering of Kaunas University of Technology. His research interests include robotics, simulation of control systems and colored Petri nets.

Olga Fiodorova received B.S., M.S and Ph.D. degrees in Electrical Engineering from Kaunas University of Technology, Kaunas, Lithuania in 2009, 2011 and 2016 respectively. She is currently vice-dean for science at Panevėžys Faculty of Technologies and Business, Kaunas University of Technology. Her research interests include computer vision, mobile robots navigation and Petri nets.

Andrius Knyš received B.S., M.S., and Ph.D. degrees in Electrical Engineering from Kaunas University of Technology, Kaunas, Lithuania, in 1998, 2000 and 2006, respectively. He is currently an associate professor at the Faculty of Electrical and Electronics Engineering, Kaunas University of Technology. His research interests include control theory and applications, transport automation.

Alma Dervinienė received B.S., M.S., and Ph.D. degrees in Electrical Engineering from Kaunas University of Technology, Kaunas, Lithuania in 1994, 1996 and 2009, respectively. She is currently an associate professor at the Faculty of Electrical and Electronics Engineering, Kaunas University of Technology. Her research interests include control theory and applications, expert systems.

Gintaras Dervinis received B.S. and M.S. degrees in Electrical Engineering and Ph.D. degree in in Informatics Engineering from Kaunas University of Technology, Kaunas, Lithuania in 1994, 1996 and 2000, respectively. He is currently an associate professor at the Faculty of Electrical and Electronics Engineering, Kaunas University of Technology. His research interests include PLC, SCADA systems, DCS systems, intelligent control and artificial intelligence systems.

Vidas Raudonis received a Ph.D. degree in Computer Science in 2010, from Kaunas University of Technology, Lithuania. He is an associate professor at the Department of Automation at the Faculty of Electrical and Electronics Engineering at Kaunas University of Technology. His research interest includes application of computation intelligence in human-computer interaction, computer vision, assistive technology and robotics, video input technologies, ANN based systems, automation systems.

Arunas Lipnickas received a B.S. and M.S. in Electrical Engineering and a Ph.D. degree in Informatics Engineering from Kaunas University of Technology, Kaunas, Lithuania, in 1996, 1998 and 2002 respectively. He is currently a professor at the Faculty of Electrical and Electronics Engineering, Kaunas University of Technology. His research interests include data analysis and image processing, robotics and automation.

Virginijus Baranauskas received B.S., M.S., and Ph.D. degrees in Electrical Engineering from Kaunas University of Technology, Kaunas, Lithuania, in 2003, 2005 and 2009, respectively. He is currently an associate professor at the Faculty of Electrical and Electronics Engineering, Kaunas University of Technology. His research interests include mobile robots navigation, Petri nets and SCADA systems.

Kastytis Kiprijonas Šarkauskas received a Ph.D. degree in Electric Measurements from Kaunas University of Technology, Lithuania, in 1970. He is a professor of the Faculty of Electrical and Electronics Engineering of Kaunas University of Technology. His research interests involve robotics, optimization, simulation of control systems and Petri nets.

Leonas Balaševičius received a Ph.D. in Informatics Engineering and Control from Kaunas University of Technology in 1999. He is currently an associate professor at the Department of Automation since 2000. The main research topics include cover optimal control, digital controllers, programmable logic controllers and SCADA systems.

References

Azarm K., & Schmidt G. (1997). *Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation*. Proceedings of the 1997 IEEE, International Conference on Robotics and Automation, pp. 3526–3534.

Baranauskas V., Derviniene A., Sarkauskas K., & Bartkevicius S. (2010). *Rationalization of the path search algorithm*. Electronics and Electrical Engineering, No. 5(101), Kaunas, Technologija, pp. 79–82, ISSN 1392-1215.

Berg J., Patil S., Sewall J., Manocha D., & Lin M. (2008). *Interactive navigation of individual agents in crowded environments*. Proceedings of ACM Symposium on I3D, pp. 139–147.

Borenstein, J., Everett, H.R., Feng, L., & Wehe, D. (1999). Mobile robot positioning & sensors and techniques. *Invited paper for the Journal of Robotic Systems, Special Issue on Mobile Robots., 14*, 231–249.

Burguera A., Gonz'alez Y., & Oliver G. (2009). *Sonar Sensor models and their application to mobile robot localization*. Sensors, Basel, pp. 10217–10243, ISSN 1424-8220.

Defoort, M., Kokosy, A., Floquet, T., Perruquetti, W., & Palos, J. (2009). Motion planning for cooperative unicycle-type mobile robots with limited sensing ranges: A distributed receding horizon approach. *Robotics and Autonomous Systems, 57*, 1094–1106.

Guo Y., & Parker L.E. (2002). *A distributed and optimal motion planning approach for multiple mobile robots*. Proceedings of the 2002 IEEE, International Conference on Robotics and Automation, Washington, pp. 2612–2619.

Jur van den Berg J., Ming L., & Manocha D. (2008). *Reciprocal velocity obstacles for real-time multi-agent navigation, robotics and automation*. Proceedings of ICRA 2008. IEEE International Conference, pp. 1928–1935.

Khashayar, R.B., & Arvin, A. (2003). Task allocation and communication methodologies for multi-robot systems. *Intelligent Automation & Soft Computing, 9*, 217–226.

Kluge B., & Prassler E. (2004). *Reflective navigation: Individual behaviors and group behaviors, Robotics and Automation*. Proceedings of ICRA '04. 2004 IEEE International Conference on Volume 4, pp. 4172–4177.

Omar, M., & Yahya, M. (2007). Mobile robot navigation using fuzzy logic. *Intelligent Automation & Soft Computing, 13*, 211–228.

Satish, P., & Madhava, K. (2007). Multi robotic conflict resolution by cooperative velocity and direction control. In Sascha Kolski (Ed.), *Mobile Robots: Perception & Navigation*. ISBN: 3-86611-283-1, InTech Available from: http://www.intechopen.com/books/mobile_robots_perception_navigation/multi_robotic_conflict_resolution_by_cooperative_velocity_and_direction_control.

Sooyong, L., & Jae-Bok, S. (2005). Mobile robot localization using range sensors: Consecutive scanning and cooperative scanning. *International Journal of Control, Automation, and Systems, 3*(1), 1–14.

Weiren, S., Kai, W., & Simon, X.Y. (2009). A fuzzy-neural network approach to multisensor integration for obstacle avoidance of a mobile robot. *Intelligent Automation & Soft Computing, 15*, 289–301.

Yi, Z., Tao, Z., & Xiaqin, L. (2012a). A new Bug-type algorithm for navigation of mobile robots in unknown environments containing moving obstacles. *Industrial Robot: An International Journal, 39*, 27–39.

Yi, Z., Tao, Z., & Xiaqin, L. (2012b). A new hybrid navigation algorithm for mobile robots in environments with incomplete knowledge. *Knowledge-Based Systems, 27*, 302–313.