Saddle Point Detection of the Navigation Function in Nonholonomic Mobile Robot Control

Wojciech Kowalczyk, Mateusz Przybyła and Krzysztof Kozłowski
Poznan University of Technology
Piotorowo 3A, Poznan, Poland
Emails: {wojciech.kowalczyk, mateusz.przybyla, krzysztof.kozlowski}@put.poznan.pl

***Abstract*—This paper presents navigation function saddle point detection and avoidance for a unicycle robot in sphere worlds. The detection is based on the gradient and Hessian analysis. The avoidance algorithm was numerically tested for two scenarios and various values of the parameter connected with the sensitivity of the saddle point detection.**

I. Introduction

Artificial potential functions (APF) are widely used in robotics since 1986 when they were proposed by Khatib [3]. The idea of repulsive and attractive interactions used to avoid collisions and move to desired position in the original version had some limitations. The main one was possibility of local minima when the APF was not designed carefully.

In the beginning of '90 Rimon and Koditschek presented concept of the navigation function. At first a sphere world version was introduced [6] which assumes that the obstacles are bounded with spheres in three dimensional spaces or with circles in planar case. Then the method was expanded to more complex environments [7], [8] and [9]. All these algorithms assumed a point-like robot without constraints.

In 2004 Urakubo [12] expanded the method introducing navigation function that takes into account nonholonomic constraints of the mobile robot. In his method the orientation of the robot can reach desired value just as both position coordinates. Convergence proof was included in the paper.

In the papers [1], [10] and [11] navigation function was used to control multiple mobile robots. They adress the problem of collision avoidance in multiagent robotic systems. In the first extension of the navigation function called multi-robot navigation functions (MRNFs) was applied. Second and third use prioritization to solve conflicts in case of concurrent goals of the agents. In work [4] authors propose a local potential function to control formation of mobile robots in an environment with obstacles.

In this paper simulation results for the algorithm presented in [12] and [13] are shown. The algorithm was tested for two scenarios with a single and two obstacles and for different values of the algorithm connected with the detection of the saddle point. The authors plan to use this method on real robot.

Section II introduces the model of the robot and the environment. Section III presents the control algorithm. In section IV saddle point detection algorithm is described. Section V presents simulation results and short comments on each of them. The final section includes concluding remarks.

II. Model of the system

The experiments were conducted on a differentially driven mobile robot with a kinematic model given by the following equation:

$$\dot{\vec{q}} = \vec{B}\vec{u}, \tag{1}$$

where vector $\vec{q}^T \triangleq [x \;\; y \;\; \theta]$ denotes the pose and x, y, θ are position coordinates and orientation of the robot with respect to a global, fixed coordinate frame. Vector $\vec{u}^T \triangleq [v \quad \omega]$ is the control vector with v denoting linear velocity and ω denoting angular velocity of the platform. Matrix $\vec{B}$ is defined as:

$$\vec{B} \triangleq \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}. \tag{2}$$

The control algorithm requires the robot task space to be bounded by a conceivable circle which defines the pool of attraction and its potential. The definition of the attracting potential function [7] is:

$$\beta_0 \triangleq \rho_0^2 - ||\vec{r} - \vec{p}_0||^2, \tag{3}$$

where ρ_0 is the radius of the task space, $\vec{r} = [x \quad y]^\top$ is the current position of the robot and $\vec{p}_0$ is the center of the task space. The controller design assumes the obstacles to be circular-shaped objects of radiuses ρ_i ($i = 1, ..., N$, N being the number of obstacles) with their centers located at positions $\vec{p}_i$. The method does not restrict the obstacles to be non-overlapping. The definition of repelling potential function for i-th obstacle is:

$$\beta_i \triangleq ||\vec{r} - \vec{p}_i||^2 - \rho_i^2. \tag{4}$$

III. The controller

Given an environment with N obstacles and a task of stabilizing the robot in its origin the total navigation potential is defined as:

$$V \triangleq \frac{C}{(C^\kappa + \beta)^{\frac{1}{\kappa}}}, \tag{5}$$

where κ is a positive, constant design parameter and

$$C \triangleq ||\vec{r}||^2 + \theta^2 \frac{k_w}{k_w + ||\vec{r}||^2}. \tag{6}$$

Symbol k_w in (6) denotes a positive, constant design parameter that allows to tune the influence of the orientation term on the navigation function depending on the Euclidean distance to the

978-1-5090-1866-6/16/$31.00 ©2016 IEEE

goal. The aggregation of obstacles repelling potential happens in term β which is defined as:

$$\beta \triangleq \prod_{i=0}^{N} \beta_i. \tag{7}$$

One must note that the iteration starts from zero, which means the inclusion of task space potential.

The control algorithm proposed in work [12] is defined as:

$$\vec{u} \triangleq -\left\{ a \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + b \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \right\} \vec{B}^T \nabla V, \tag{8}$$

where a is a positive, constant design parameter and

$$b \triangleq -\bar{b} \frac{\vec{L}^\top \nabla V}{h(g)}. \tag{9}$$

Symbol $\bar{b}$ in (9) denotes a positive, constant design parameter and $\vec{L}^\top \triangleq [\sin\theta \quad -\cos\theta \quad 0]$. Function $h(g)$ is defined as:

$$h(g) \triangleq g^2 + \epsilon\sqrt{g}, \tag{10}$$

where $g \triangleq ||\vec{B}^\top \nabla V||$ and ϵ_g is a small positive constant. Finally, ∇V denotes the gradient of the navigation function with respect to variables x, y and θ. Regardless of number of obstacles, the gradient can be obtained in analytical form as:

$$\nabla V = \frac{\nabla C (C^\kappa + \beta)^{\frac{1}{\kappa}} - \frac{C}{\kappa}(C^\kappa + \beta)^{(\frac{1}{\kappa}-1)}(\kappa C^{\kappa-1}\nabla C + \nabla\beta)}{(C^\kappa + \beta)^{(\frac{2}{\kappa})}}, \tag{11}$$

where

$$\nabla C = \begin{bmatrix} \frac{\partial C}{\partial x} \\ \frac{\partial C}{\partial y} \\ \frac{\partial C}{\partial \theta} \end{bmatrix}^\top = \begin{bmatrix} 2x(1 - \frac{k_w \theta^2}{(k_w + ||\vec{r}||^2)^2}) \\ 2y(1 - \frac{k_w \theta^2}{(k_w + ||\vec{r}||^2)^2}) \\ 2\theta \frac{k_w}{k_w + ||\vec{r}||^2} \end{bmatrix}^\top \tag{12}$$

and

$$\nabla\beta = \sum_{i=0}^{N} \left\{ \frac{\partial \beta_i}{\partial \vec{q}} \prod_{j=0, j\neq i}^{N} \beta_j \right\}. \tag{13}$$

As noted in [7] all the undesired local minima of navigation function (5) disappear as the parameter κ increases. An algorithm for automatically tuning analytic navigation functions for sphere worlds was presented in [2]. The tuning parameter must satisfy a lower bound to ensure convergence to the desired value. In this paper navigation functions have been manually tuned to assure convergence. For the sufficently high value of the κ parameter navigation function (5) has a critical point associated with each isolated obstacle, the saddle point. V has no other critical points other then these points. Saddle points are unstable equilibrium points. In [12] special control procedure for saddle point avoidance is described. It uses time varying trigonometric function to push the robot away from the unstable equilibrium point.

IV. Saddle point detection

The detection of the saddle point can be based on analysis of Hessian of the navigation function $H(V)$. If $\nabla V = \vec{0}$ (in practice $||\nabla V|| < \epsilon_\nabla$, where ϵ_∇ - small value). In section V detailed analysis of the influence of this parameter on the process of saddle point detection and avoidance is presented. The saddle point occurs if one eigenvalue of $H(V)$ is negative.

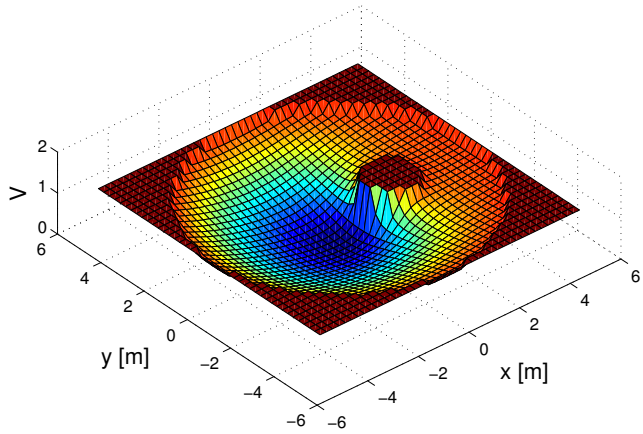

Fig. 1: Navigation function cross-section for $\theta = 0$

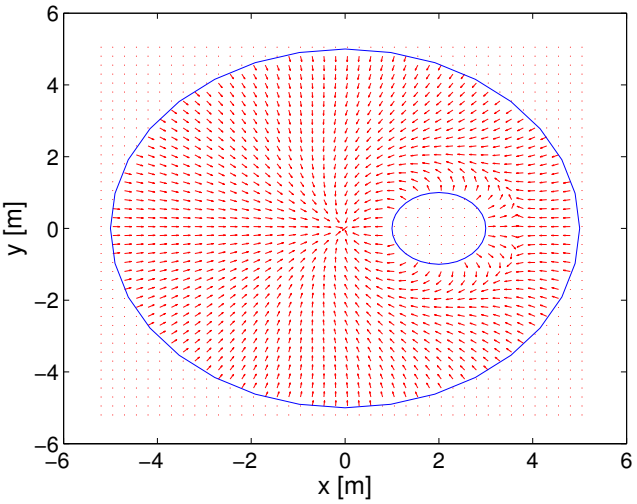

Fig. 2: Vector field cross-section $(-\nabla V)$ for $\theta = 0$

Hessian of the navigation function is as follows:

$$H(V) = \frac{\partial(\nabla V)}{\partial \vec{q}} = \frac{H(C)}{(\beta + C^\kappa)^{1/\kappa}} + \\ - \frac{C(H(\beta) + \kappa C^{\kappa-1} H(C) + \kappa C^{\kappa-2} \nabla C^\top \nabla C(\kappa - 1))}{\kappa(\beta + C^\kappa)^{\frac{1}{\kappa}+1}} \\ - \frac{2(\nabla\beta + \kappa C^{\kappa-1}\nabla C)^\top \nabla C}{\kappa(\beta + C^\kappa)^{\frac{1}{\kappa}+1}} \tag{14}$$

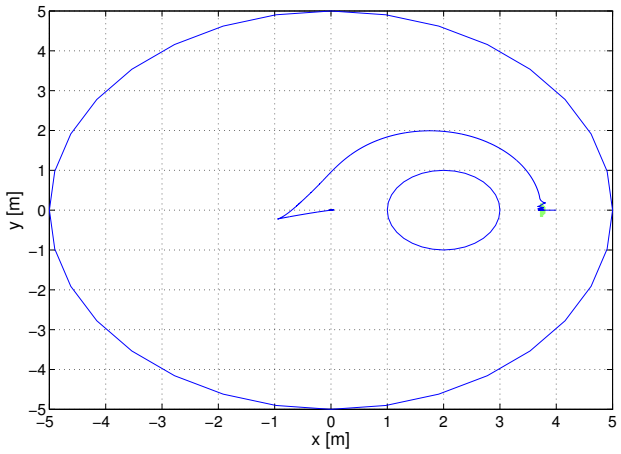

Fig. 3: Robots path for $\epsilon_\nabla = 0.005$

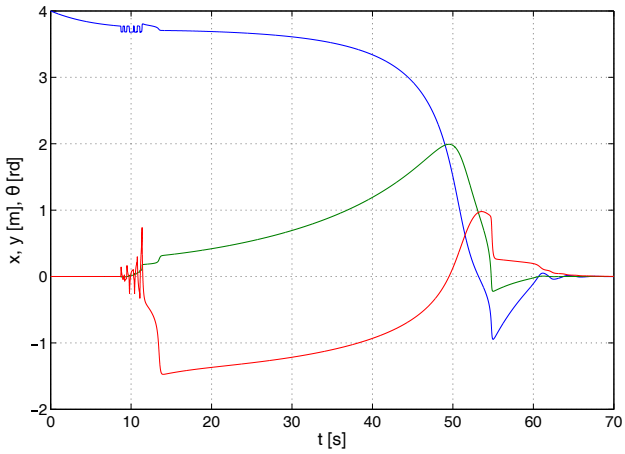

Fig. 5: Position and orientation coordinates for $\epsilon_\nabla = 0.005$; x - blue, y - green θ - red

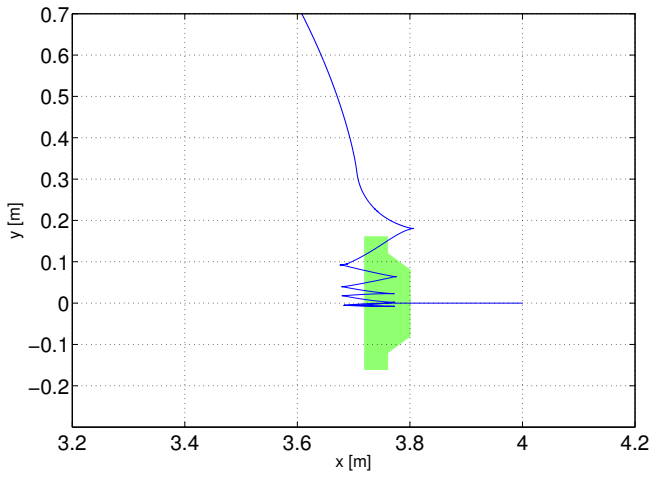

Fig. 4: Robots path for $\epsilon_\nabla = 0.005$ - initial part

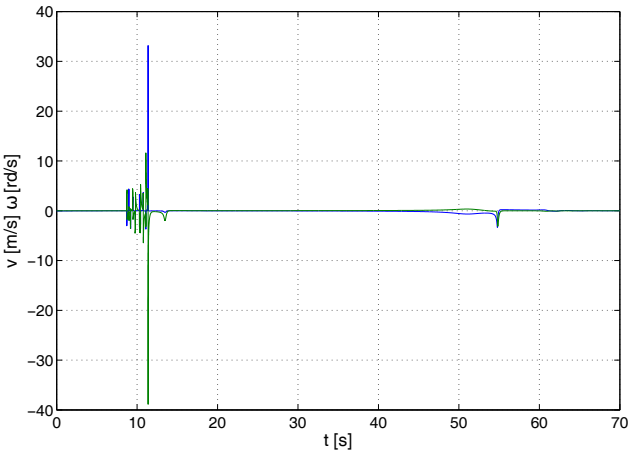

Fig. 6: Control values for $\epsilon_\nabla = 0.005$; v - blue, ω - green

$$+\frac{C(\frac{1}{\kappa}+1)(\nabla\beta+\kappa C^{\kappa-1}\nabla C)^\top(\nabla\beta+\kappa C^{\kappa-1}\nabla C)}{\kappa(\beta+C^\kappa)^{\frac{1}{\kappa}+2}}.$$

The Hessian of the attraction function is given with equation:

$$H(C)=\frac{\partial(\nabla C)}{\partial\vec{q}}=\begin{bmatrix}\frac{\partial^2 C}{\partial x^2} & \frac{\partial^2 C}{\partial x\partial y} & \frac{\partial^2 C}{\partial x\partial\theta}\\ \frac{\partial^2 C}{\partial y\partial x} & \frac{\partial^2 C}{\partial y^2} & \frac{\partial^2 C}{\partial y\partial\theta}\\ \frac{\partial^2 C}{\partial\theta\partial x} & \frac{\partial^2 C}{\partial\theta\partial y} & \frac{\partial^2 C}{\partial\theta^2}\end{bmatrix} \tag{15}$$

where

$$\frac{\partial^2 C}{\partial x^2}=\frac{8k_w\theta^2x^2}{(||\vec{r}||^2+k_w)^3}-\frac{2k_w\theta^2}{(||\vec{r}||^2+kw)^2}+2 \tag{16}$$

$$\frac{\partial^2 C}{\partial x\partial y}=\frac{\partial^2 C}{\partial y\partial x}=\frac{8k_w\theta^2xy}{(||\vec{r}||^2+k_w)^3} \tag{17}$$

$$\frac{\partial^2 C}{\partial x\partial\theta}=\frac{\partial^2 C}{\partial\theta\partial x}=-\frac{4k_w\theta x}{(||\vec{r}||^2+k_w)^2} \tag{18}$$

$$\frac{\partial^2 C}{\partial y^2}=\frac{8k_w\theta^2y^2}{(||\vec{r}||^2+kw)^3}-\frac{2k_w\theta^2}{(||\vec{r}||^2+kw)^2}+2 \tag{19}$$

$$\frac{\partial^2 C}{\partial y\partial\theta}=\frac{\partial^2 C}{\partial\theta\partial y}=-\frac{4k_w\theta y}{(||\vec{r}||^2+k_w)^2} \tag{20}$$

$$\frac{\partial^2 C}{\partial\theta^2}=\frac{2k_w}{||\vec{r}||^2+k_w} \tag{21}$$

The Hessian of the obstacle function is as follows:

$$H(\beta)=\frac{\partial(\nabla\beta)}{\partial\vec{q}}= \tag{22}$$

$$= \sum_{i=0}^{N} \left(\frac{\partial^2 \beta_i}{\partial \vec{q}^2} \prod_{j=0, j\neq i}^{N} \beta_j + \frac{\partial \beta_i}{\partial \vec{q}} \sum_{k=0}^{N} \left\{ \frac{\partial \beta_k}{\partial \vec{q}} \prod_{j=0, j\neq i, j\neq k}^{N} \beta_j \right\} \right).$$

If the saddle point is detected the avoidance procedure is activated. The control given by Eq. (8) is temporarily replaced with the following one [12]:

$$\vec{u_a} \triangleq [a_1 sin(wt) + b_1 \;\; a_2 sin(wt)]^T \tag{23}$$

where a_1, a_2, w and b_1 are chosen such that $[b_1 \;\; \frac{a_1 a_2}{2w} \;\; 0]^T$ is parallel to the eigenvector of $H(V)$ of the negative eigenvalue and $|\frac{a_1}{w}| \ll 0$, $|\frac{a_2}{w}| \ll 0$ and $|b_1| \ll 0$.

V. Simulation results

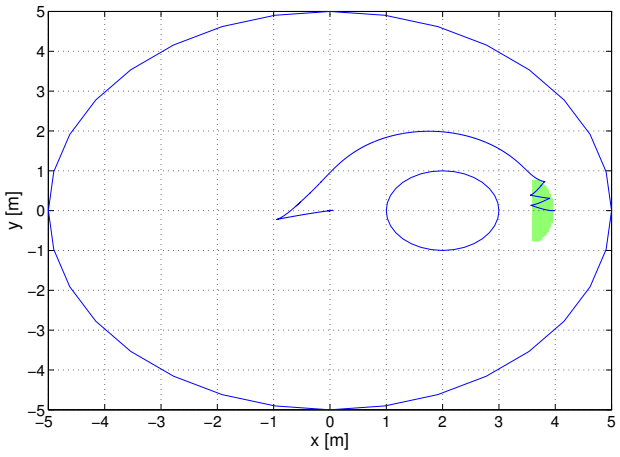

Fig. 7: Robots path for $\epsilon_\nabla = 0.02$

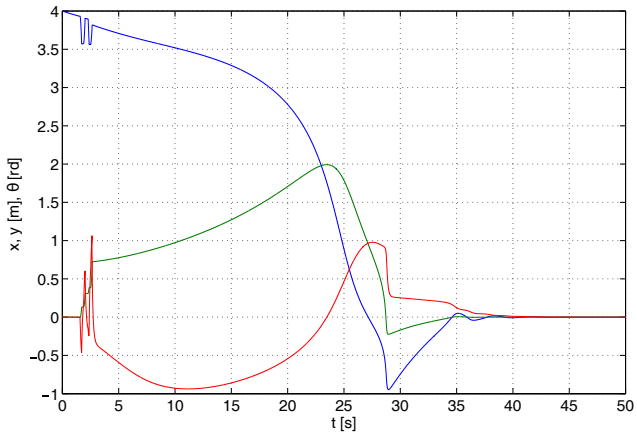

Fig. 8: Position and orientation coordinates for $\epsilon_\nabla = 0.02$; x - blue, y - green θ - red

An example for the single obstacle described by the obstacle function $\beta_1 = ||[x \;\; y]^\top - [2 \;\; 0]^\top||^2 - 1$ and the world constrained by the zero-th obstacle $\beta_0 = 25 - ||[x \;\; y]^\top||^2$. The gradient and Hessian of the internal obstacle are as follows:

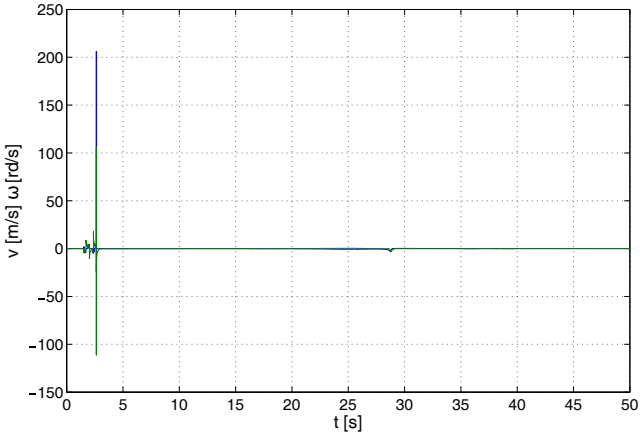

Fig. 9: Control values for $\epsilon_\nabla = 0.02$; v - blue, ω - green

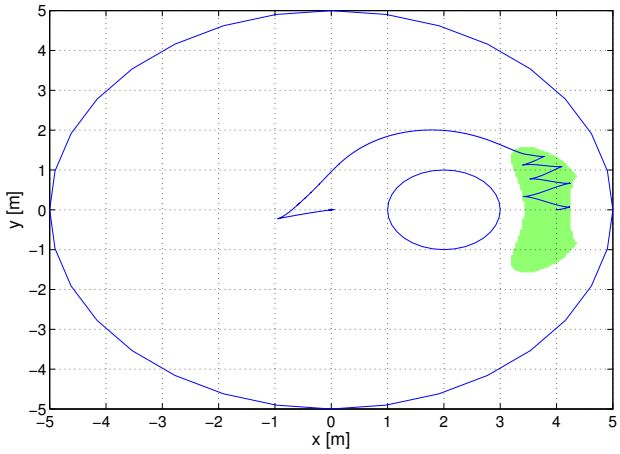

Fig. 10: Robots path for $\epsilon_\nabla = 0.05$

$\nabla\beta_1 = [2([x \;\; y] - [2 \;\; 0]) \;\; 0]^\top$ and $H(\beta_1) = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. The gradient and Hessian of the zero-th obstacle are as follows: $\nabla\beta_0 = [-2([x \;\; y]) \;\; 0]^\top$ and $H(\beta_0) = -\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. It is worth to note that for star shape obstacles [8] the same approach can be applied. In this case only the formulas of β_i and $\nabla\beta_i$ must be modified. In Figs. 1 and 2 cross-section of the navigation function and $-\nabla$ for $\theta = 0$ are shown. As the total navigation function is three-dimensional (x, y, θ) only cross-sections for particular value of one of the space variable can be visualized. The saddle point exists for $x = 3.75$ and $y = 0$. The parameters of the algorithm are as follows: $\kappa = 2$, $k_w = 1$, $a = 2$, $b = 10$, $\epsilon_g = 10^{-6}$. They were tuned manually to get proper behaviour and meet the constraints described in [12] (a, b) and [8] (κ).

In Fig. 3 the robots path in (x, y) plane is shown. In all presented simulations desired coordinates are given by the vector $[0 \; 0 \; 0]^\top$. Initial position of the robot is $(4, 0)$. As these

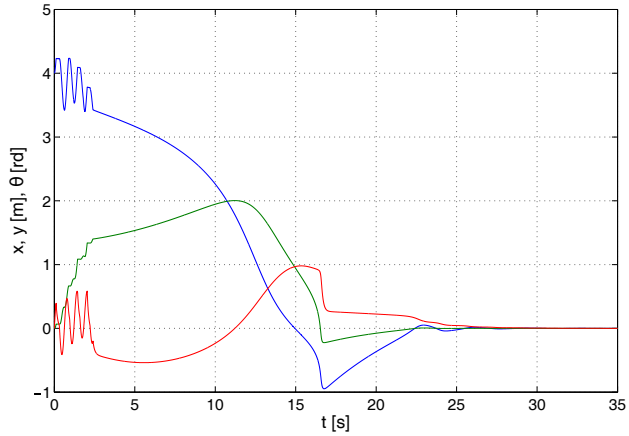

Fig. 11: Position and orientation coordinates for $\epsilon_\nabla = 0.05$; x - blue, y - green θ - red

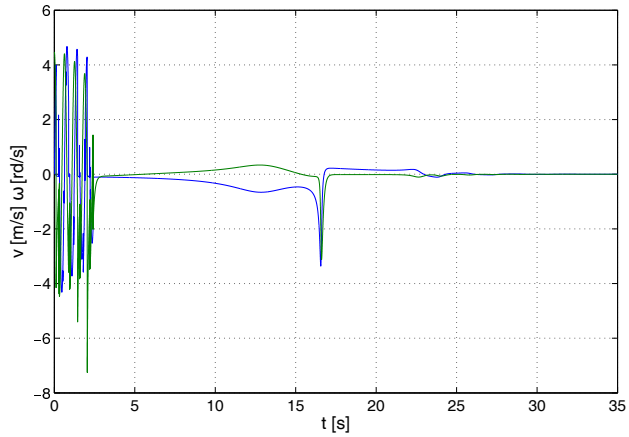

Fig. 12: Control values for $\epsilon_\nabla = 0.05$; v - blue, ω - green

coordinates are in the stable manifold of the saddle point the system converges to it. The value of the ϵ_∇ was set to the very small value 0.005. It causes that the controller detects the saddle point only in its close neighborhood. The detection area is marked with green spot. In Fig. 4 initial part of the path is shown. When the saddle point is detected robot starts to move in the direction indicated by the eigenvector of the $H(V)$ of the negative eigenvalue. In Fig. 5 time graphs of the position and orientation coordinates are presented. The values converge to zero (the desired value). After $70s$ the module of the error is less then 10^{-5}. In Fig. 6 control for linear and angular velocities respectively are shown.

In Fig. 7 the same scenario was used to test the system for the greater value of $\epsilon_\nabla = 0.02$. This causes that the saddle point avoidance is activated in greater area in the neighborhood of the saddle point. In presented case the detection area was very close to the initial position of the robot. This configuration resulted in quicker activation of the saddle point avoidance procedure. In Fig. 8 one can observe that the desired values are reached in less then $45s$ (much faster in comparison to the

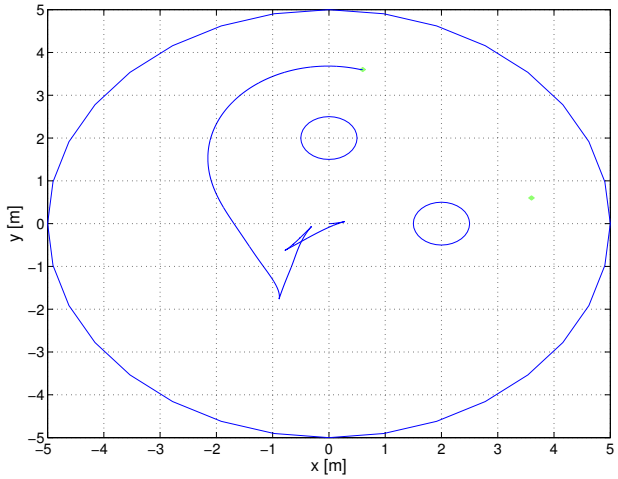

Fig. 13: Robots path for two obstacles in the environment

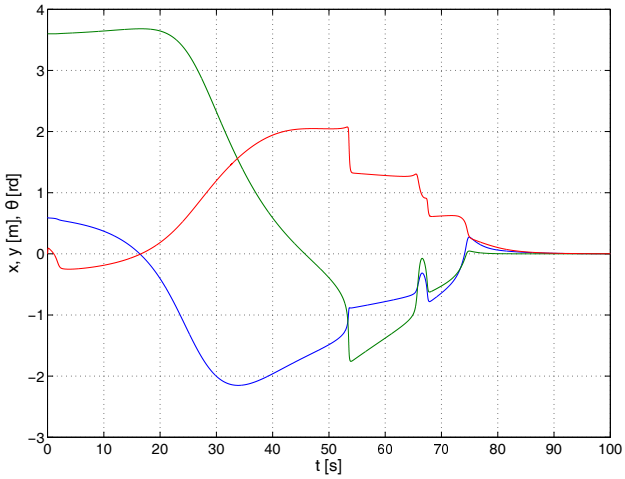

Fig. 14: Position and orientation coordinates for two obstacles in the environment; x - blue, y - green θ - red

previous case). In Fig. 9 control signals are shown.

In Fig. 10 robots path for $\epsilon_\nabla = 0.05$ is presented. In this case the detection area is the largest comparing to the previous cases. As one can observe in Fig. 11 position and orientation coordinates converge to desired values in about $28s$. Fig. 12 shows control signals.

In Fig. 13 the algorithm is tested for the environment with two obstacles located in $(2, 0)$ and $(0, 2)$. The radius of the obstacles is 0.5. There are two saddle points (each of them associated with one internal obstacle) in $(0.6, 3.6)$ and $(3.6, 0.6)$. Initial position of the robot was set to $(0.6, 3.6)$. In Fig. 14 coordinates of the robot as a function of time are shown. They converge to the near zero value in about 100s. In Fig. 15 liner and angular velocity controls are presented. In Figs. 16 and 17 cross-section of the navigation function and $-\nabla$ for $\theta = 0$ are presented.

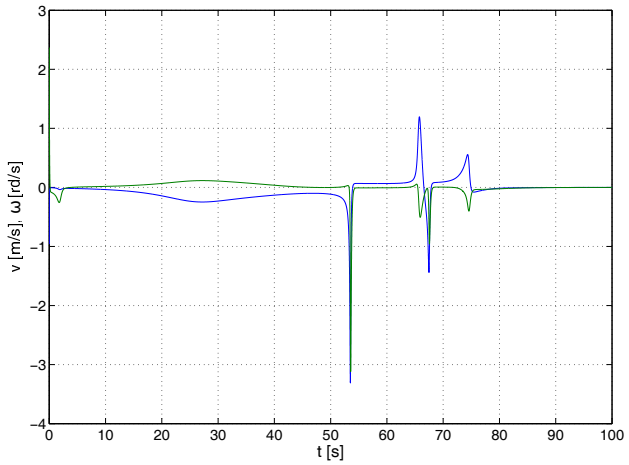

Fig. 15: Control values for two obstacles in the environment; v - blue, ω - green

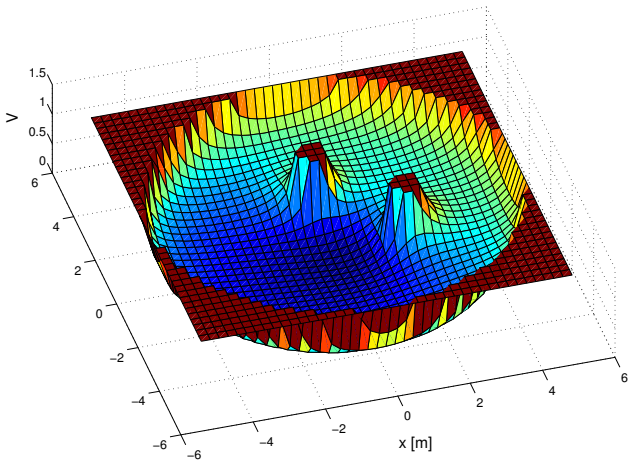

Fig. 16: Navigation function cross-section for $\theta = 0$

The shapes, numbers and relative locations of the obstacles affect the transient states of the system, however, functionality connected with the saddle points is not affected by these factors.

VI. Conclusion

In this paper saddle point detection and avoidance algorithm was investigated. It was shown that by increasing the area surrounding the saddle point where avoidance procedure is activated may significantly reduce convergence time. The authors interest in presented subject is due to need of use in real robotic system. In previous experiments navigation function method was investigated for the scenarios without need to saddle point avoidance [5]. Next research will focus on experimental tests of the saddle point detection and avoidance.

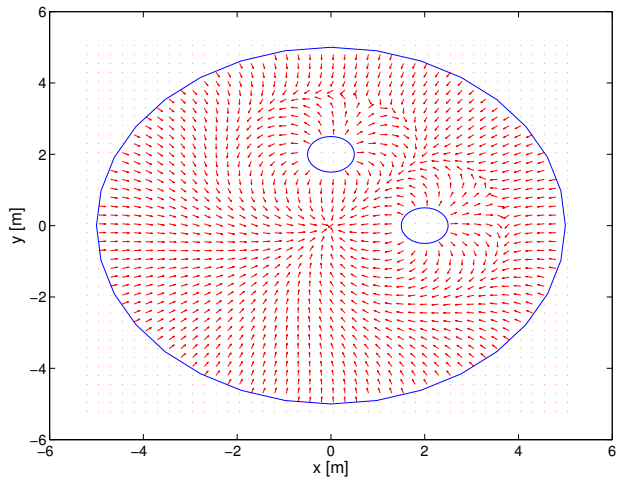

Fig. 17: Vector field cross-section $(-\nabla V)$ for $\theta = 0$

References

[1] D.V. Dimarogonasa, S.G. Loizoua, K.J. Kyriakopoulos and M.M. Zavlanosb, A feedback stabilization and collision avoidance scheme for multiple independent non-point agents, Automatica, Vol. 42, No. 2, pp. 229–243, 2005.

[2] I. Filippidis and K.J. Kyriakopoulos, Adjustable navigation functions for unknown sphere worlds, IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), pp. 4276–4281, 2011.

[3] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, The International Journal of Robotics Research, Vol 5, No. 1, pp. 90–98, 1986.

[4] W. Kowalczyk, K. Kozlowski, and J.K. Tar, Trajectory tracking for multiple unicycles in the environment with obstacles, International Conference on Robotics in Alpe-Adria-Danube Region (RAAD), pp. 451–456, 2010.

[5] W. Kowalczyk, M. Przybyla, K. Kozlowski, Control of a mobile robot and collision avoidance using navigation function-experimental verification, Robot Motion and Control (RoMoCo), 2015 10th International Workshop on, pp. 148-152. x

[6] E. Rimon and D.E. Koditschek, Exact robot navigation using cost functions: the case of distinct spherical boundaries, IEEE International Conference on Robotics and Automation, Vol. 3, pp. 1791–1796, 1988.

[7] E. Rimon and D. Koditschek, Exact Robot Navigation Using Artificial Potential Functions, IEEE Transactions on Robotics and Automation, Vol 8, No 5, pp. 501–518, 1992.

[8] E. Rimon and D. E. Koditschek, The construction of analytic diffeomorphisms for exact robot navigation on star worlds, Transaction of the American Mathematical Society, Vol. 327, pp. 71–116, 1991.

[9] E. Rimon and D.E. Koditschek, Exact robot navigation using artificial potential fields, IEEE Transactions on Robotics and Automation, Vol. 8, No. 5, pp. 501–518, 1992.

[10] G. Roussos, K.J. Kyriakopoulos, Decentralized and Prioritized Navigation and Collision Avoidance for Multiple Mobile Robots, Distributed Autonomous Robotic Systems - Springer Tracts in Advanced Robotics, Vol. 83, pp. 189–202, 2013.

[11] G. Roussos, K.J. Kyriakopoulos, Completely Decentralised Navigation of Multiple Unicycle Agents with Prioritisation and Fault Tolerance, IEEE Conference on Decision and Control (CDC), pp. 1372–1377, 2010.

[12] T. Urakubo, K. Okuma and Y. Tada, Feedback control of a two wheeled mobile robot with obstacle avoidance using potential functions, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vol. 3, pp. 2428–2433, 2004.

[13] T. Urakubo, Feedback Stabilization of a Nonholonomic System with Potential Fields: Application to a Two-wheeled Mobile Robot among Obstacles, Nonlinear Dynamics, Vol. 81, Issue 3, pp. 1475-1487, 2015.