Motion Planning and Trajectory Tracking on 2-D Manifolds embedded in 3-D Workspaces*

Xanthi Papageorgiou, Savvas G. Loizou and Kostas J. Kyriakopoulos

Control Systems Laboratory, M.E. Dept.
National Technical University of Athens
Athens, Greece
{*xpapag,sloizou,kkyria*}*@mail.ntua.gr*

Abstract— In this paper we present a methodology that drives and stabilizes a robotic agent moving in a three dimensional environment, to a 2-dimensional manifold embedded in the workspace. Once the agent reaches the manifold, depending on the application, it performs a motion planning or a trajectory tracking task. Appropriately constructed belt-zone vector fields guarantee that the agent will not depart the 2-D manifold proximity area, while carrying out the motion planning or trajectory tracking task. The derived closed form feedback control law guarantees global convergence and collision avoidance. The properties of the proposed algorithm are verified through non-trivial computer simulations.

Index Terms— Navigation, tracking, belt-zones.

I. INTRODUCTION

One of the main tasks of neuro-robotics is to make a robot execute a task by interfacing with the neural system (e.g. by processing electromyographic activity, etc.). In most of the cases, these signals are noisy and rather "incomprehensible" to directly control a robot particularly in clattered environments. In those cases we need a strategy to make the robot compliant with its environment and at the same time avoiding obstacles. The methodology presented in this paper can be applied as well to the case of autonomous robotic surface painting, cleaning, inspection, etc. The main difficulties of the above tasks arise when the considered surface is not planar. Moreover the non-planar surface might include "bad regions" that must be avoided.

Most of previous relevant research has focused on the problem of automotive painting of surfaces that are convex and have no holes, [1], [2], [3]. Also in [1], the used approach decomposes the coverage trajectory generation problem into three subproblems: selection of the start curve, selection of the speed profiles along each pass, and selection of the spacing between the passes.

In our work, we are using navigation functions, [4], [5], [6], [9], [10], to drive the agent safely to a surface non-modeled in the navigation function. Once the agent is in the close proximity of the surface, a second controller takes over to stabilize the agent at a predefined distance from the surface (which partly depends on the surface curvature), while at the same time, depending on the application, performs motion planning and trajectory tracking task across the surface. Our basic idea, is to use a sliding

mode controller [12], based on the belt-zone concept [7], to stabilize the system on the surface and at the same time with appropriate construction of the neighboring vector fields driven to carry out the motion planning or trajectory tracking task. The main contribution of this paper can be summarized as follows:

- A novel theoretically guaranteed trajectory tracking controller, achieving obstacle avoidance and concurrent stabilization during tracking over 2-D manifolds embedded in 3-D workspaces.
- A provably correct way to perform motion planning and concurrent stabilization tasks over 2-D manifolds embedded in 3-D workspaces.

The rest of the paper is organized as follows: Section II introduces preliminary definitions, notation and some technical Lemmas, required for further discussion. Section III describes the construction of the vector field that is used for robot navigation while section IV presents the proposed control law. Section V presents the simulation results and the paper concludes with section VI.

II. PRELIMINARIES

Our analysis is demonstrated by considering the navigation problem of a moving holonomic agent in 3-D workspace. Let the admissible configuration space (workspace) for the robot be $\mathcal{W} \subset \mathbb{R}^3$. The obstacle free subset of the workspace is denoted $\mathcal{W}_{free} \subseteq \mathcal{W}$, and $\varphi : \mathcal{W} \to \mathbb{R}$ is the potential (navigation) function which models the environment. Let $\mathcal{O} \in \mathcal{W} \setminus \mathcal{W}_{free}$ be the i'th obstacle, $i = 1, \ldots n_O$, where n_O is the number of obstacles. We define q_{d1} as the navigation function's target configuration. q_{d1} exists in the closure of a 3-D closed surface, which surface is defined below.

Define a vector valued C^2 function: $g(s_1, s_2) = [g_x(s_1, s_2), g_y(s_1, s_2), g_z(s_1, s_2)]^T : \mathbb{R} \times \mathbb{R} \to \mathcal{R}(g)$ representing a closed surface. The range $\mathcal{R}(g) \subset \mathcal{W}_{free}$ of the function will serve as the boundary of the surface over which the surface processing task will take place. We can now define the following topology (Fig. 1):

1) The surface's internal, J^-.
2) The surface's boundary, ∂g.
3) The surface's external, J^+.

The tangent vectors on the surface w.r.t parameters s_1

419

Proc. IEEE Int'l Conf. Robotics & Automation (ICRA-2005), Barcelona, Spain, April 2005. p.p 419-424

and s_2 can be defined as:

$$g_{s_1}(s_1, s_2) = \frac{\partial g(s_1, s_2)}{\partial s_1} = \left[\frac{\partial g_x}{\partial s_1}, \frac{\partial g_y}{\partial s_1}, \frac{\partial g_z}{\partial s_1}\right]^T \quad (1)$$

$$g_{s_2}(s_1, s_2) = \frac{\partial g(s_1, s_2)}{\partial s_2} = \left[\frac{\partial g_x}{\partial s_2}, \frac{\partial g_y}{\partial s_2}, \frac{\partial g_z}{\partial s_2}\right]^T \quad (2)$$

Since $(g_{s_1} \times g_{s_2}) \neq 0$, $\forall s_1, s_2 \in \mathbb{R}$, the vectors g_{s_1}, g_{s_2} are linear independent everywhere. Therefore, every tangent vector to the surface is a linear combination of these two vectors g_{s_1} and g_{s_2}, (Fig. 1).

Also, we can define a normalized perpendicular vector to the surface as: $N = \frac{g_{s_1} \times g_{s_2}}{\|g_{s_1} \times g_{s_2}\|}$.

We define as $\nabla g(s_1, s_2)$ the tangent vector on the surface directed across the maximum curvature direction (see Appendix). Furthermore, we define the perpendicular vector to the surface, as $\nabla g^\perp(s_1, s_2) = (g_{s_1} \times g_{s_2})$.
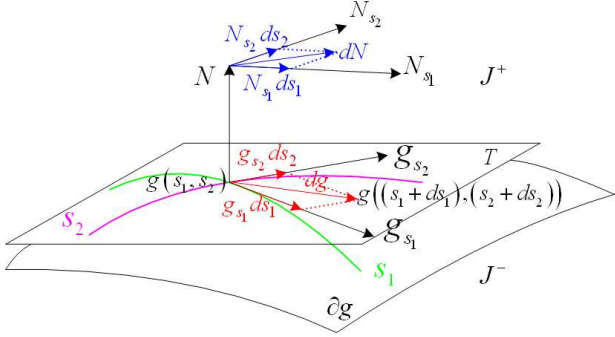

Fig. 1. Representation of tangent's and perpendicular's vectors, of a surface, variations w.r.t. parameter's modification.

We now define the vector valued function

$$a(s_1, s_2) = g(s_1, s_2) + \rho \cdot N(s_1, s_2) \quad (3)$$

It holds that: $da = dg + \rho dN \Rightarrow dg^T \cdot da = dg^T \cdot dg + \rho(dg^T \cdot dN)$, where $dg = g_{s_1} \cdot ds_1 + g_{s_2} \cdot ds_2$, $dN = N_{s_1} \cdot ds_1 + N_{s_2} \cdot ds_2$, and ds_1, ds_2 are length variations across the boundary (Fig. 1). Consequently, we have that the radius of curvature is equal to:

$$\rho = \frac{1}{\kappa_n}\left(1 - \frac{dg^T \cdot da}{dg^T \cdot dg}\right) \quad (4)$$

where κ_n is the perpendicular curvature of the surface, which is equal to (see Appendix):

$$\kappa_n = \frac{-dg^T \cdot dN}{dg^T \cdot dg} \quad (5)$$

Following the same line of thought as in [7], we need the function $a(s_1, s_2)$, from (3) to be bijective. This is guaranteed locally if we require that $da \neq 0$, $\forall dg \neq 0 \Leftrightarrow ds_1, ds_2 \neq 0$. So if we have $da = 0$, from (4), the radius of $g(s_1, s_2)$ at the point (s_1, s_2) is the inverse of the curvature of $g(s_1, s_2)$:

$$\rho = \frac{1}{\kappa_n} \quad (6)$$

This implies that locally, in order to satisfy the needed condition of the function $a(s_1, s_2)$, we must choose a $\rho > 0$ such that:

$$\rho < \rho_m = \min_{s_1, s_2}\left|\frac{1}{\kappa_n}\right| = \frac{1}{\kappa^*} \quad (7)$$

This justifies the selection of the maximum curvature, so a non-negative ρ can always be defined.

Let us now assume that the function $\varphi : \mathcal{W}_{free} \to \mathbb{R}$ is a navigation function [10], q_{d1} the target configuration for the navigation function and $g(s_1, s_2)$ a vector valued bijective C^2 function, defining a closed surface in $\mathcal{W}$, homeomorphic to a sphere, not modeled in the navigation function. The solutions of the equation $\dot{x} = -\nabla\varphi$ are absolutely continuous and using standard topological arguments, for $q_{d_1} \in J^-$ and initial conditions $x_0 \in J^+$, those solutions intersect the surface $g(s_1, s_2)$.

III. NAVIGATION VECTOR FIELD

A. Belt Zones

A predefined navigation function is implemented to drive the robot to the closed surface defined by $g(s_1, s_2)$ containing the NF's destination point q_{d1} in it's internal. Since our goal is to perform motion tasks across the boundary of $g(s_1, s_2)$, when the robot reaches a certain distance from the boundary of $g(s_1, s_2)$, an appropriate task (navigation or trajectory tracking) specific vector field starts influencing the robot.

To construct such a vector field we are going to use the concept of sliding motion along surfaces of discontinuity. This type of motion is generated by two neighboring vector fields, which are attached across the boundary ∂g of the surface. The region where those vector fields are defined is called the "belt zone", and these vector fields, "belt zone" vector fields, [7]. The "belt zone" is the region close to the ∂g and is thought to be composed of an "internal belt" and an "external belt" region. For the motion tasks considered in this paper the widths of the internal and external belt regions are consider to be fixed. The "belt zone" concept is represented in Fig. 2.

Let us define the vector functions which are used to describe the belt zones to which we have referred above.

$$\beta(s_1, s_2) = g(s_1, s_2) + \delta_1 \cdot N \quad (8)$$

$$\gamma(s_1, s_2) = \beta(s_1, s_2) + \delta_2 \cdot N \quad (9)$$

with $0 < (\delta_1 + \delta_2) < \rho_m$. Our goal is to stabilize the robot on the middle surface defined by (8).

Now we are in position to define the sets of "internal region", $\mathcal{E}$, and the "external region", $\mathcal{I}$.

$$\mathcal{I} = \{q : q = (1 - \lambda) \cdot \beta(s_1, s_2) + \lambda \cdot g(s_1, s_2)\} \quad (10)$$

$$\mathcal{E} = \{q : q = (1-\lambda) \cdot \beta(s_1, s_2) + \lambda \cdot \gamma(s_1, s_2), \lambda \neq 0\} \quad (11)$$

where $\lambda \in [0, 1]$, (Fig. 2).

B. Belt zone vector fields

We wish to choose the vector fields which steers the agent to slide across the surface $\beta(s_1, s_2)$.

Let $h_\mathcal{E} \in \mathcal{E}$, $h_\mathcal{I} \in \mathcal{I}$ and (s_1^o, s_2^o) belong to the domain of g. Then let h_o be such that $h_o = g(s_1^o, s_2^o)$. Then for each $h_\mathcal{E}$ and $h_\mathcal{I}$, there exists unique h_o hence unique (s_1^o, s_2^o) such that:

$$h_\mathcal{E} = (1 - \lambda_1)\beta(s_1^o, s_2^o) + \lambda_1 \gamma(s_1^o, s_2^o) \quad (12)$$

$$h_\mathcal{I} = (1 - \lambda_2)\beta(s_1^o, s_2^o) + \lambda_2 g(s_1^o, s_2^o) \quad (13)$$

for some λ_1, $\lambda_2 \in [0, 1]$ as long as $(\delta_1 + \delta_2) < \rho_m$.
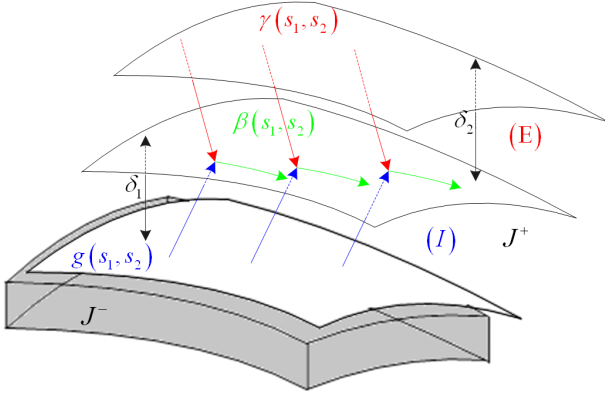

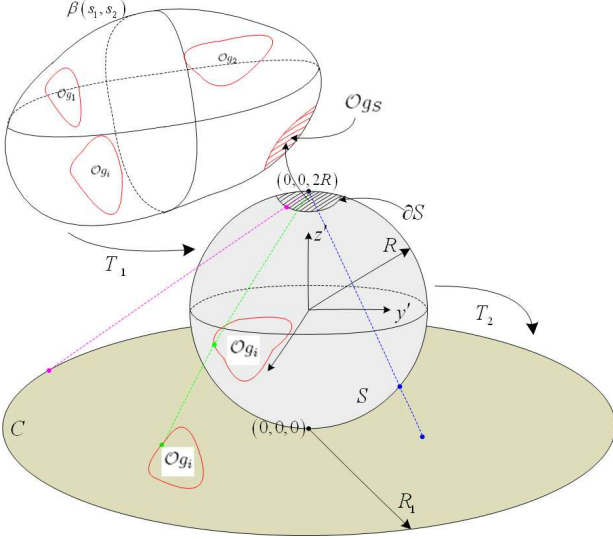Fig. 2. Representation of Belt Zones, in a part of a surface.

Fig. 3. Transformation of a surface, firstly, on a sphere and then into a disc.

This follows from $g(s_1, s_2)$ being bijective, and since (7) holds, we can always choose nonzero δ_1 and δ_2 such that $\delta_1 + \delta_2$ is smaller than the surface's minimum radius. Since the distances of the belt zone boundaries are lower than the local radius of $g(s_1, s_2)$ for each (s_1, s_2), then the set of points of minimum distance of any $h_\mathcal{E} \in \mathcal{E}$ or $h_\mathcal{I} \in \mathcal{I}$ from surface β contain only one point: $\beta(s_1^o, s_2^o)$ of this surface. So we can define the surjective functions

$h_o^\mathcal{E} : q \in \mathcal{E} \rightarrow q \in \beta(s_1, s_2)$ and $h_o^\mathcal{I} : q \in \mathcal{I} \rightarrow q \in \beta(s_1, s_2)$. Let $(s_1, s_2)(q) = \beta^{-1}(q)$ denote the inverse function of β, and $(s_1^\mathcal{E}, s_2^\mathcal{E}) = (s_1, s_2)(h_o^\mathcal{E}(q))$, and $(s_1^\mathcal{I}, s_2^\mathcal{I}) = (s_1, s_2)(h_o^\mathcal{I}(q))$.

We choose for $q \in \mathcal{E}$ (external zone) the vector field:

$$V_\mathcal{E} = -k_0 \cdot \nabla \varphi_\tau(s_1^\mathcal{E}, s_2^\mathcal{E}) + k_2(\lambda_1) \cdot \nabla g^\perp(s_1^\mathcal{E}, s_2^\mathcal{E}) \quad (14)$$

where k_0 is positive tuning constant, and $-k_2$ is a class-$\mathcal{K}$ function of λ_1. This means that at the beginning the influence of the perpendicular vector field is high, and when the robot approaches the β surface this influence is going to zero to avoid chattering effects. $\nabla \varphi_\tau(s_1, s_2)$ is a task specific vector field defined on the β surface.

For the region $q \in \mathcal{I}$ we choose the vector field:

$$V_\mathcal{I} = -k_0 \cdot \nabla \varphi_\tau(s_1^\mathcal{I}, s_2^\mathcal{I}) + k_4(\lambda_2) \cdot \nabla g^\perp(s_1^\mathcal{I}, s_2^\mathcal{I}) \quad (15)$$

where k_4 is a class-$\mathcal{K}$ function of λ_2, for the same reason as k_2.

C. Vector fields on a surface

Once the robot enters the belt zone, it will remain there as the boundaries of the belt zone are repulsive and it will execute a sliding motion along the surface $\beta(s_1, s_2)$, depending on φ_τ.

Now it is necessary to define the appropriate vector field φ_τ on the surface, that will be used in order the robot to navigate across the surface to a specified destination point. We can transform the closed surface $\beta(s_1, s_2)$ to the boundary of a sphere S with radius R. We know that the expression of the surface in polar coordinates is:

$$r = \sqrt{\beta_x^2(s_1, s_2) + \beta_y^2(s_1, s_2) + \beta_z^2(s_1, s_2)} \quad (16)$$

Assuming that $s_1 = \vartheta$ and $s_2 = \phi$, with $\vartheta \in [0, 2\pi)$, and $\phi \in [0, \pi]$, we have that the ϑ, ϕ, can be used for the sphere

$$T_1 = id_\beta \quad (17)$$

where id_A is the identity map of a set A, and we have changed the radius, i.e. the cartesian coordinates transformation can be expressed as $q' = \frac{R}{r} \cdot q$, where $q = [x \quad y \quad z]^T$. The obstacles on the surface $\beta(s_1, s_2)$ have the form $\mathcal{O}_{g_i}(s_1, s_2) = 0$, with $i = 1, \ldots M$, the number of the obstacles, (Fig. 3). This representation of the obstacles is very useful because it does not need transformation on the sphere, when s_1, s_2 are longitude and latitude angles, since in this case the transformation affects only the polar distance.

The next step is to project (stereographic projection) the surface of the sphere S to a disc C, which belong to the plane $z = 0$. Our goal is to construct a vector field on a manifold, but a necessary condition for this is that we have a manifold with boundary. The stereographic projection of the singular point is at infinity. So we need to avoid the singular point. It is known that a vector field on a sphere must have at least one singular (critical) point, [13]. In our case the critical point is $(0, 0, 2R)$. In order to avoid this point, and to have a boundary on the manifold, we choose a region on the top of the sphere which is repulsive

for $\phi < \phi_o$, and $\vartheta \in [0, 2\pi)$. This region is transformed to the surface as a virtual obstacle $\mathcal{O}g_S$, (Fig. 3). The transformation $T_2 : S \to C$, is defined for $\vartheta \in [0, 2\pi)$, $\phi \in [\phi_o, \pi]$ as:

$$\begin{bmatrix} x \\ y \end{bmatrix} = T_2 = \frac{2 \cdot R \cdot \sin \phi}{1 - \cos \phi} \begin{bmatrix} \cos \vartheta \\ \sin \vartheta \end{bmatrix} \tag{18}$$

The obstacles, after this projection are modeled as functions of x, y. They compose star-shaped sets, and we can change their coordinates into a disc, which can be done with analytic diffeomorphisms [9]. So we have the diffeomorphism h_λ from a star-shaped set into a disc. The problem is thus reduced to designing a navigation function φ_N on a disk shaped workspace [9] and pulling the navigation vector field back to the initial surface. The navigation function $\tilde{\varphi}$ on the surface $\beta(s_1, s_2)$ is thus defined as:

$$\tilde{\varphi} = \varphi_N \circ h_\lambda \circ T_2 \circ T_1 \tag{19}$$

and the task specific vector field is $\nabla \varphi_\tau = \nabla \tilde{\varphi}$ for the navigation task.

IV. CONTROL STRATEGY

We assume that we have a stationary environment and the robot can be described trivially by a fully actuated, first order kinematic model. In the workspace, the robot is represented by a point. The obstacles present in the environment are modeled by the navigation function, [9]. The goal is for the robot to be able to navigate using the navigation function constructed on a 3-D space, to execute sliding motion on the surface $\beta(s_1, s_2)$, to complete a task on the surface, avoiding obstacles.

A. Reaching a point on a surface

Assume that the robot's initial configuration is $x(0) \in J^+$, and we would like it move towards the surface, in order to reach a specific point on it (e.g. for the neuro-robotic set-up, a manipulator wants to place an object on a table, without breaking it). So, it is necessary to define the control law with which we can achieve this.

Let us define a vector field through the following multifunction:

$$f(q) = \begin{cases} -\nabla \varphi(q), & q \in \mathcal{W}_{ext} \\ V_{\mathcal{E}}(q), & q \in \mathcal{E} \\ V_{\mathcal{I}}(q), & q \in \mathcal{I} \end{cases} \tag{20}$$

where $\mathcal{W}_{ext} = \{\mathcal{W}_{free} \bigcap J^+\} \setminus \{\mathcal{E} \bigcup \mathcal{I}\}$

Consider the following differential equation:

$$\dot{x} = f \tag{21}$$

Proposition 1: The system $\dot{x} = u$ under the control law $u = f$ with f as defined in (20) is globally asymptotically stable, almost everywhere[1].

Proof: We will consider the system as operating in two possible modes: mode Φ where $q \in \mathcal{W}_{ext}$ and mode $\mathcal{B}$ where $q \in \mathcal{E} \bigcup \mathcal{I}$.

[1]i.e. everywhere except a set of initial conditions of measure zero.

Since φ is a navigation function, all initial states in the original environment are brought to the origin, except a set of initial states having measure zero, that lead to unstable saddle points. For $q \in \Phi$ we have that: $\dot{\varphi}(x) = -\|\nabla \varphi\|^2 \overset{a.e.}{<} 0$, where $\dot{\varphi}(x) = 0 \; \forall x \in \{0\} \bigcup \mathcal{S}$ and $\mathcal{S}$ being the set of measure zero of saddle points. A navigation function has at least as many isolated saddle points as the number of obstacles, [10]. Hence under the given control law the potential is strictly decreasing almost everywhere in Φ. As stated in a previous section any system trajectory starting in J^+ will enter the belt zone. Upon entrance in the belt zone the system switches to mode $\mathcal{B}$. We can now form the following Lyapunov function: $V_1 = \tilde{\varphi} + \delta_N^2$, where δ_N is the distance of a point from the β surface. Then, $\dot{V}_1 = -k_0 \|\nabla \tilde{\varphi}\|^2 - 2\delta_N k_i \|\nabla g^\perp\|^2 \overset{a.e.}{<} 0$, since $sign(k_i) = sign(\delta_N)$, $i \in \{2, 4\}$. Thus the system will stabilize on the β surface while performing the navigation task encoded in $\tilde{\varphi}$. ∎

The following states that when the robot enters the belt zone, it can never leave it.

Corollary 1: The set $\{\mathcal{E} \bigcup \mathcal{I}\} \subset \mathcal{W}$ is positively invariant.

Proof: Assume that the robot is on the boundary of the belt zone, that is $q \in \partial \{\mathcal{E} \bigcup \mathcal{I}\}$. The perpendicular vector towards the inner of the set is $-\nabla g^\perp(s_1^{\mathcal{E}}, s_2^{\mathcal{E}})$, and $\nabla g^\perp(s_1^{\mathcal{I}}, s_2^{\mathcal{I}})$ for the region $\mathcal{E}$ and $\mathcal{I}$, respectively. Then it is true that $-\nabla g^\perp(s_1^{\mathcal{E}}, s_2^{\mathcal{E}}) \cdot f > 0$ and $\nabla g^\perp(s_1^{\mathcal{I}}, s_2^{\mathcal{I}}) \cdot f > 0$. Consequently, the boundary of the set $\{\mathcal{E} \bigcup \mathcal{I}\}$ is repulsive. ∎

B. Tracking

Now let us consider a tracking task across the surface $\beta(s_1, s_2)$. The task is described by a known trajectory $q_d(t)$ on it (e.g. for neuro-robotic set-up, a manipulator keep a pen and we would like it to write something on a paper). Let us now define the appropriate control law in order to track the predefined trajectory.

We introduce a navigation function of the form:

$$\varphi_\kappa(q, t) = \frac{\gamma_d(q, t)}{[\gamma_d^\kappa(q, t) + \beta(q, t)]^{1/\kappa}} \tag{22}$$

where γ_d is the distance to the trajectory, and $\beta(q)$ is the product of obstacle functions, [9].

We consider convergence of the system to a small ball of radius $\varepsilon > 0$ containing the target.

Before defining the control we need some preliminary definitions. We define by $\nabla^2 \varphi(q, t)$ the Hessian of the φ. Let $\lambda_{\min}$, λ_{max} be the minimum and the maximum eigenvalues of the Hessian, $\hat{v}_{\lambda_{\min}}$, $\hat{v}_{\lambda_{\max}}$ the unit eigenvectors corresponding to the minimum and maximum eigenvalues of the Hessian. Then the P_1 region, introduced in [8], can be used to identify sets of points that contain measure zero sets whose positive limit sets are saddle points:

$$P_1 = (\lambda_{\min} < 0) \wedge (\lambda_{\max} > 0) \wedge (|\hat{v}_{\lambda_{\min}} \cdot \nabla V| < \varepsilon_1)$$

where $\varepsilon_1 < \min\limits_{C = \{q : \|q - q_d\| = \varepsilon\}} (\|\nabla \varphi(C)\|)$. If $|\hat{v}_{\lambda_{\min}} \cdot \nabla V| = 0$ then the set P_1 consists of the

measure zero set of initial conditions that lead to saddle point, [8]. In view of this, ε_1 can be chosen to be arbitrarily small so the sets defined by P_1 eventually consist of thin sets containing sets of initial conditions that lead to saddle points.

Now consider that the system is operating in mode $\mathcal{B}$ defined in the proof of Proposition 1.

Proposition 2: System (21) in mode $\mathcal{B}$, under the control law defined by the task specific vector field:

$$\nabla\varphi_\tau = \nabla\bar{\varphi} + \frac{\partial\bar{\varphi}}{\partial t} \cdot \frac{\nabla\bar{\varphi}}{sw\left(\|\nabla\bar{\varphi}\|^2, \varepsilon^2\right) - \varepsilon^2 \cdot \sigma\left(\frac{\partial\bar{\varphi}}{\partial t}\right) \cdot \sigma\left(\|\nabla\bar{\varphi}\|^2\right)} \tag{23}$$

converges to the set where $\|q - q_d\| < \varepsilon$, almost every-where[1], with $\sigma(x) = \frac{x}{1+|x|}$, $sw(x,e) = \begin{cases} x, & x \geq e \\ e, & x < e \end{cases}$, and $\bar{\varphi} = \varphi_\kappa(q,t) \circ h_\lambda \circ T_2 \circ T_1 + \delta_N^2$, where T_1, T_2 come from equations (17), (18) respectively.

Proof: We form the following Lyapunov function:

$$V = \bar{\varphi}(q,t) \tag{24}$$

and take it's derivative:

$$\dot{V} = \frac{\partial V}{\partial t} + f \cdot \nabla V \tag{25}$$

After substituting the control law (23) and since we pursue convergence in the set $\|q - q_d\| < \varepsilon$, we get:

$$\dot{V} = -\|\nabla V\|^2 + \frac{\partial V}{\partial t} \cdot \left(1 - \frac{\|\nabla V\|^2}{\|\nabla V\|^2 - \varepsilon^2 \cdot \sigma\left(\frac{\partial V}{\partial t}\right) \cdot \sigma\left(\|\nabla V\|^2\right)}\right) -$$
$$- 2\delta_N k_i \left\|\nabla g^\perp\right\|^2 \leq -\|\nabla V\|^2 +$$
$$+ \frac{\partial V}{\partial t} \cdot \left(1 - \frac{\|\nabla V\|^2}{\|\nabla V\|^2 - \varepsilon^2 \cdot \sigma\left(\frac{\partial V}{\partial t}\right) \cdot \sigma\left(\|\nabla V\|^2\right)}\right)$$

since the same arguments from the proof of Proposition 1 hold w.r.t. $\delta_N k_i$. We can now discriminate the following cases:

- $\frac{\partial V}{\partial t} = 0 \Rightarrow \dot{V} = -\|\nabla V\|^2 \leq 0$
- $\frac{\partial V}{\partial t} > 0 \Rightarrow 0 < \sigma(\frac{\partial V}{\partial t}) < 1 \Rightarrow \|\nabla V\|^2 - \varepsilon^2 < \|\nabla V\|^2 - \varepsilon^2 \cdot \sigma\left(\frac{\partial V}{\partial t}\right) \cdot \sigma\left(\|\nabla V\|^2\right) < \|\nabla V\|^2 \Rightarrow \dot{V} \leq 0$
- $\frac{\partial V}{\partial t} < 0 \Rightarrow -1 < \sigma(\frac{\partial V}{\partial t}) < 0 \Rightarrow \|\nabla V\|^2 < \|\nabla V\|^2 - \varepsilon^2 \cdot \sigma\left(\frac{\partial V}{\partial t}\right) \cdot \sigma\left(\|\nabla V\|^2\right) < \|\nabla V\|^2 + \varepsilon^2 \Rightarrow \dot{V} \leq 0$

The sets defined by P_1 are by construction repulsive. We assume that the system's initial conditions are in the set $\{\mathcal{E} \bigcup \mathcal{I}\} \setminus \mathcal{P}$, where the set $\mathcal{P} = \{q : \|\nabla V\| < \varepsilon_1\}$. ∎

Remark 1: In practice we can choose an ε_1, such that $\varepsilon_1 < \min\{\varepsilon_0, \|\nabla V(q_0, t_0)\|\}$, so we can be sure that the system's initial conditions are not in $\mathcal{P}$.

V. SIMULATION RESULTS

Computer simulations have been carried out to verify the feasibility and efficacy of the proposed methodology. The scenario of the simulation contains two 3-D (ellipsoid) obstacles centered at $\mathcal{O}_1 : (-0.55, 0.6, 0)$ and $\mathcal{O}_2 : (-0.7, 0, 0)$. The agent's initial configuration was: $q(0) = (-0.7, 0.65, 0)$ and the first target was set at $q_{d1} = (0, 0, 0)$. The surface's obstacles shown in Fig. 4-6, are the intersections of ellipsoid obstacles with the surface. They are centered at $\mathcal{O}_{g1} : (-0.3, 0.2, 0.2121)$, $\mathcal{O}_{g2} : (0.3, 0.2, 0.2121)$ and $\mathcal{O}_{g3} : (-0.3, -0.2, -0.2121)$ and the second target was set at $q_{d2} = (0.3, -0.2, -0.2121)$. Also,

there is an extra obstacle on the surface which comes from the constraint of $\phi \geq 5°$ at the sphere's vector field, which is centered at $\mathcal{O}_{gs} : (0, 0, 0.3)$. After the agent reaches its destination point (q_{d2}), it starts a tracking task, to track a predefined trajectory which is the yellow colored line in Fig. 4-6. Also, each of these figures depict the simulation results, from a different point of view, since there is a 3-D view simulation. Our algorithm successfully converges to the goal configuration and track the predefined trajectory avoiding obstacles.

A second simulation have been performed in order to check, that when the trajectory is passing through an obstacle, the robot can avoid it by leaving the trajectory until this is out of the obstacle again. The results are shown in Fig. 7.
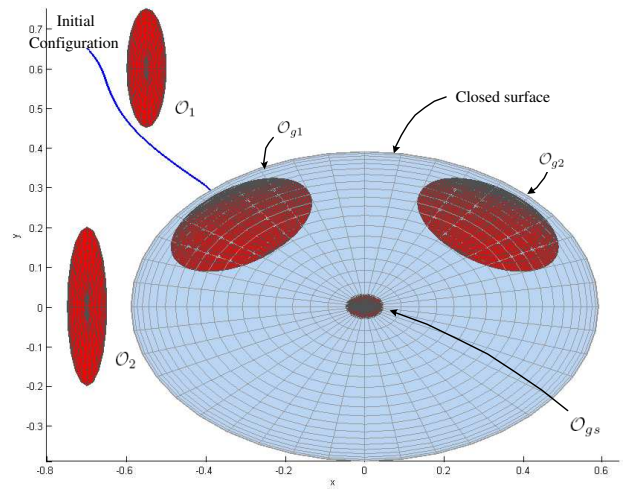
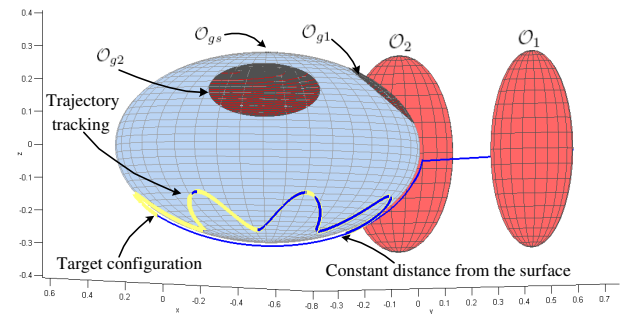

Fig. 4. Simulation results 1: top view.

Fig. 5. Simulation results 1: reaching a point on the surface.

VI. CONCLUSION

A methodology of performing navigation and tracking tasks over a 2-dimensional manifold embedded in a 3-dimensional workspace was derived. After safely navigating the robot to the 2-D manifold, task specific vector fields direct the robot towards accomplishing a navigation or a trajectory tracking task across the 2-D manifold. The methodology has theoretically guaranteed global convergence and collision avoidance properties. Due to the

closed form of the feedback controller, the methodology is particularly suitable for implementation on real time systems with limited computation capability.

Further research includes considering surface properties in the construction of the belt zone vector fields and implementing the methodology to real neuro-robotic systems taking into account their dynamics and kinematic constraints.
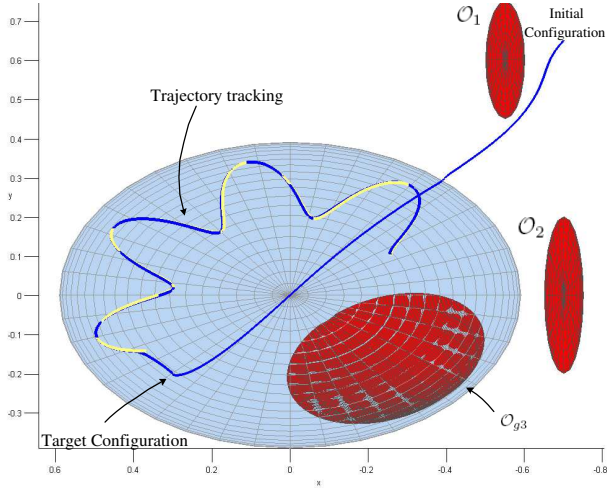

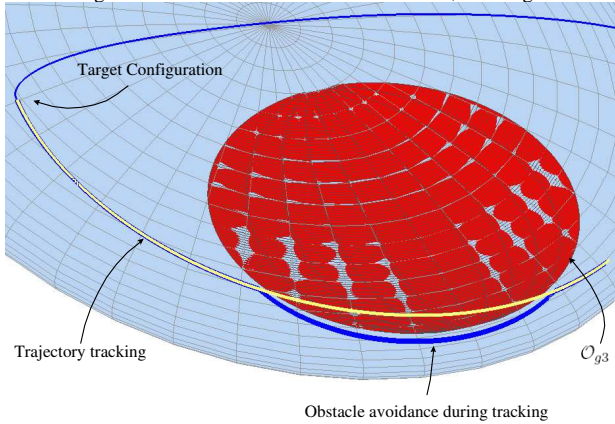Fig. 6. Simulation results 1: bottom view, tracking.

Fig. 7. Simulation results 2: bottom view, tracking.

REFERENCES

[1] P.N. Atkar, D.C. Conner, A. Greenfield, H. Choset, A. A. Rizzi, "Uniform Coverage of Simple Surfaces Embedded in $\mathbb{R}^3$ for Auto-Body Painting", Carnegie Mellon University, Pittsburgh, 2004.

[2] P.N. Atkar, H. Choset, A. A. Rizzi, "Towards Optimal Coverage of 2-Dimensional Surfaces Embedded in $\mathbb{R}^3$: Choice of Start Curve", Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems, Las Vegas, 2003.

[3] D.C. Conner, P.N. Atkar, A. A. Rizzi, H. Choset, "Deposition Modeling for Paint Application on Surfaces Embedded in $\mathbb{R}^3$", Technical Report, Carnegie Mellon University, Pittsburgh, 2002.

[4] H.G. Tanner, S.G. Loizou, K.J. Kyriakopoulos, "Nonholonomic Navigation and Control of Cooperating Mobile Manipulators", IEEE Transactions on Robotics and Automation, vol 19, no 1, pp 53-64, February 2003.

[5] S.G. Loizou, K.J. Kyriakopoulos, "Closed Loop Navigation for Multiple Non-Holonomic Vehicles", International Conference on Robotics and Automation, 2003.

[6] S.G. Loizou, K.J. Kyriakopoulos, "Closed Loop Navigation for Multiple Holonomic Vehicles", IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002.

[7] S.G. Loizou, H.G. Tanner, V. Kumar, K.J. Kyriakopoulos, "Closed Loop Motion Planning and Control for Mobile Robots in Uncertain Environments", Proceedings of the 42nd IEEE Conference on Decision and Control, Maui, Hawaii USA, 2003.

[8] S.G. Loizou, D.V. Dimarogonas, K.J. Kyriakopoulos, "Decentralized Feedback Stabilization of Multiple Nonholonomic Agents", (Invited Paper) Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA, 2004.

[9] E. Rimon and D.E. Koditschek, "Exact robot navigation using artificial potential functions", IEEE Trans. on Robotics and Automation, vol. 8, no. 5, p.p. 501-518, 1992.

[10] D.E. Koditschek, E.Rimon, "Robot navigation functions on manifolds with boundary", Advances Appl. Math., 11:412442, 1990.

[11] A. Filippov, "Differential Equations with Discontinuous Righthand Sides", Kluwer Academic Publishers, 1988.

[12] V.I. Utkin, J. Guldner, J. Shi, "Sliding Mode Control in Electromechanical Systems", Taylor & Francis, 1999.

[13] W.M. Boothby, "An introduction to differentiable manifolds and Riemannian geometry", Academic Press, London, 1986.

[14] H.K. Khalil, "Nonlinear Systems", Prentice-Hall, 1996.

APPENDIX

A C^2 surface $g = g(s_1, s_2)$, on Euclidian space $\mathbb{E}^3$ is defined univocally, from two invariable quantities, which are functions of ds_1, ds_2, where ds_1, ds_2 are length variations across the boundary. These quantities are the first and the second fundamental forms, respectively: $I = dg \cdot dg$, $II = -dg \cdot dN$ with $dg = (g((s_1 + ds_1), (s_2 + ds_2)) - g(s_1, s_2)) \Rightarrow dg = g_{s_1} \cdot ds_1 + g_{s_2} \cdot ds_2$ where dN is defined as following, (Fig. 1): $dN = \frac{\partial N}{\partial s_1} \cdot ds_1 + \frac{\partial N}{\partial s_2} \cdot ds_2 \Rightarrow dN = N_{s_1} \cdot ds_1 + N_{s_2} \cdot ds_2$.

Writing all the above in a matrix form we have that: $\mathbf{I} = J^T \cdot J$, $\mathbf{II} = J^T \cdot H$ where $J = [g_{s_1} \quad g_{s_2}]$, and $H = [N_{s_1} \quad N_{s_2}]$, with $N_{s_1} = \frac{\partial N}{\partial s_1}$ and $N_{s_2} = \frac{\partial N}{\partial s_2}$.

The perpendicular curvature of the surface can be defined from the inner product of the curvature vector and the normalized perpendicular vector to the surface: $\kappa_n = k \cdot N \Leftrightarrow \kappa_n = \frac{II}{I} \Leftrightarrow$ or in a matrix form: $\mathbf{S} = \mathbf{I}^{-1} \cdot \mathbf{II}$.

Consider the eigenvector problem $\mathbf{S} \cdot \mathbf{p} = \kappa \cdot \mathbf{p}$. Each eigenvector $\mathbf{p}$ is a principal direction. The corresponding eigenvalue κ is a principal curvature. The vector $\mathbf{p}$ is an 2-vector given in terms of tangent space coordinates.

Let us define: $\mathbf{E} = g_{s_1}^T \cdot g_{s_1}$, $\mathbf{F} = g_{s_1}^T \cdot g_{s_2}$, and $\mathbf{G} = g_{s_2}^T \cdot g_{s_2}$. Also we can define: $\mathbf{L} = -g_{s_1}^T \cdot N_{s_1}$, $\mathbf{M} = -\frac{1}{2}(g_{s_1}^T \cdot N_{s_2} + g_{s_2}^T \cdot N_{s_1})$, and $\mathbf{N} = -g_{s_2}^T \cdot N_{s_2}$.

A direction $\frac{ds_1}{ds_2}$ is a principal direction and a real valued number κ is a principal curvature which correspond to this direction in a point of the surface, if and only if, in this point, the following relations hold:

$$\left. \begin{array}{l} (\mathbf{L} - \kappa \mathbf{E})ds_1 + (\mathbf{M} - \kappa \mathbf{F})ds_2 = 0 \\ (\mathbf{M} - \kappa \mathbf{F})ds_1 + (\mathbf{N} - \kappa \mathbf{G})ds_2 = 0 \end{array} \right\} \quad (26)$$

with $ds_1^2 + ds_2^2 \neq 0$. Accordingly we have that: $(\mathbf{EG} - \mathbf{F}^2)\kappa^2 - (\mathbf{EN} + \mathbf{GL} - 2\mathbf{FM})\kappa + (\mathbf{LN} - \mathbf{M}^2) = 0$. This means that we have minimum and maximum curvature κ_1, κ_2. As will be shown subsequently, we will need to use the maximum curvature: $\kappa^* = max(\kappa_1, \kappa_2)$. From the above equation (26), we have that the direction of the maximum curvature, is $\mu = \frac{ds_1^*}{ds_2^*}$. When $\kappa_1 = \kappa_2$ then we have only one direction to choose from equation (26).