Robust source-seeking hybrid controllers for nonholonomic vehicles[*]

Christopher G. Mayhew, Ricardo G. Sanfelice, and Andrew R. Teel †

***Abstract*— We develop a hybrid controller that drives a nonholonomic vehicle to the source of a radiation-like signal. The control signal depends only on measurements of the signal, which may be corrupted by noise, and internal states of the controller. The control strategy is inspired by a line minimization-based algorithm for unconstrained optimization of nonlinear functions without gradient information. We state the main properties of the algorithm and then discuss its implementation in a hybrid controller that coordinates the optimization algorithm with vehicle steering. We present semi-global practical convergence and stability results for the closed-loop system. These results are illustrated by simulation and experiment.**

I. Introduction

In this paper we examine the problem of steering an autonomous vehicle to the source of a radiation-like signal. We assume that only measurements of the signal are available and that they may be corrupted by noise. The radiation signal may be thought of as a potential function in the vehicle's environment, taking a maximum (or minimum) value at its "source." Hence, one could view this as an unconstrained optimization problem, where a vehicle must be steered to collect measurements for the optimization routine.

This problem has received much attention in the recent literature, with approaches ranging from gradient descent with a single vehicle [5] or multiple vehicles [3] to executing a simplex-inspired optimization algorithm with a team of two vehicles [4]. Strategies using *extremum seeking* were also suggested for nonholonomic vehicles in [15], [6].

In this paper, we consider steering a constant-velocity nonholonomic vehicle with a bounded angular rate to the source. We extend the strategy proposed by Mayhew et al. in [11] to the "Dubins vehicle," state a global convergence result for a class of functions, and illustrate this convergence with experimental results and simulations. We begin with a precise statement of the problem and vehicle model.

II. Problem Statement and Notation

We consider the problem of navigating an autonomous vehicle to the source of a scalar radiation-like signal existing in the environment. We assume that this signal is described by a continuously differentiable function $\varphi : \mathbb{R}^2 \to \mathbb{R}$. Letting $\operatorname{rge}\varphi = \{y \in \mathbb{R} : y = \varphi(x)\}$, we assume that for every $c \in \operatorname{rge}\varphi$, $L_\varphi(c) = \{x \in \mathbb{R}^2 : \varphi(x) \leq c\}$ is compact, that φ has a unique global minimum of $\varphi(x^*)$ and $\nabla\varphi(x) = 0$ if and only if $x = x^*$.

We assume that prior knowledge of the function is not available; instead, only noise-corrupted measurements of the function are available. Additionally, these measurements can only be taken at the current position of the vehicle. As a further constraint inspired by autonomous vehicles in GPS-denied environments, we do not assume that measurements of vehicle position are available.

In this paper, we use a common nonholonomic vehicle model that has a state (x, θ), where $x = (x_1, x_2) \in \mathbb{R}^2$ and $\theta \in \mathbb{R}$ denote the absolute position and heading of the vehicle. The state evolves according to

$$\dot{x} = \gamma \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \qquad \dot{\theta} = \omega, \tag{1}$$

where γ is a positive constant defining forward velocity and ω is a control input for angular velocity, constrained by $|\omega| \leq \gamma/\rho$, where ρ is the minimum turning radius. This nonholonomic vehicle model is commonly referred to as the *Dubins vehicle* [7] in the literature. In this setting, the objective is design a control system that steers the vehicle, from any initial condition, to a neighborhood of x^*.

Additionally, we make the following notational conveniences. The set $\mathbb{S}^{n-1} = \{v \in \mathbb{R}^n : \|v\|_2 = 1\}$ denotes the unit 2-sphere in $\mathbb{R}^n$. Given $v \in \mathbb{S}^n$, the vector $v_\perp \in \mathbb{S}^n$ denotes the unit-vector orthogonal to v, i.e., $v^T v_\perp = 0$. The set $\mathbb{Z}_{>0}$ denotes the set of integers greater than zero, i.e., $\{1, 2, \ldots\}$. We denote by $\mathcal{Q}_n$, the set of all functions $f : \mathbb{R}^n \to \mathbb{R}$ such that $f = f^* + (x - x^*)^\top P(x - x^*)$ with $P = P^\top > 0$. We denote the closed unit ball in $\mathbb{R}^n$ centered at $\xi \in \mathbb{R}^n$ of radius ϵ as $\xi + \epsilon\mathbb{B} = \{y \in \mathbb{R}^n : |\xi - y| \leq \epsilon\}$. For a given function $\Gamma : X \to X$, we denote Γ^n as the composition of Γ n times, that is,

$$\Gamma^n = \underbrace{\Gamma \circ \ldots \circ \Gamma}_{n \text{ times}}.$$

Finally, we denote a vector rotation of α radians in $\mathbb{R}^2$ by the rotation operator $\mathcal{R}_\alpha$.

III. Robust Optimization & Vehicle Steering: A Hybrid Harmony

We propose a control scheme that coordinates an unconstrained optimization algorithm with vehicle path planning to solve the source-seeking problem posed in Section II. Our optimization algorithm, presented in Section III-A, stems from the ideas of Smith [14], Powell [12], and Rosenbrock [13], who utilized conjugate directions to minimize functions of several variables. Through vehicle path planning,

† {mayhew,teel}@ece.ucsb.edu, Center for Control Engineering and Computation, Electrical and Computer Engineering Department, University of California, Santa Barbara, CA 93106-9560.

sricardo@mit.edu, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, MA 02139, Tel (617) 324-8498,

*Research partially supported by the National Science Foundation under Grant no. CCR-0311084, Grant no. ECS-0622253, and Grant no. ECS-0242798 and by the Air Force Office of Scientific Research under Grant no. F9550-06-1-0134.

it is possible to collect function measurements for such an algorithm. As we describe here (and as shown in [11] for a point-mass vehicle without nonholonomic constraints), such a synergy is possible with a hybrid controller. This is a natural approach, as the vehicle is an inherently continuous-time system, while an optimization algorithm is an inherently discrete-time system.

The following discussion relies on the notion of a line minimization, which is the essential component of our optimization algorithm. Given a function $\varphi : \mathbb{R}^n \to \mathbb{R}$, a point $x^0 \in \mathbb{R}^n$, and a non-zero vector $v \in \mathbb{R}^n$, a *line minimization* from x^0 in the direction of v consists of finding the value $\lambda \in \mathbb{R}$ such that $x^0 + \lambda v$ is the unique minimum of φ along v from x^0. We denote the operation of performing a line minimization from a point $x^0 \in \mathbb{R}^n$ along a direction $v \in \mathbb{R}^n$ on a function φ and returning the value λ as a mapping $\mu_\varphi : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$. This operation is equivalently known as a "line search."

A. The Recursive Smith-Powell Algorithm (RSP)

Given $x^0 \in \mathbb{R}^2$ and $v \in \mathbb{S}^1$,

Step 1) Calculate $x^{*1} = x^0 + \mu_\varphi(x^0, v)v$.
Step 2) Calculate $x^{*2} = x^{*1} + \mu_\varphi(x^{*1}, v_\perp)v_\perp$.
Step 3) Calculate $x^{*3} = x^{*2} + \mu_\varphi(x^{*2}, v)v$.
Step 4) Update the direction v with the vector $v^+ = \Phi(\mu_\varphi(x^{*1}, v_\perp)v_\perp, \mu_\varphi(x^{*2}, v)v, v)$,
Step 5) Set $x^0 = x^{*3}$ and go to **Step 1)**.

The function $\Phi : \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{S}^1 \to \mathbb{S}^1$ updates v by

$$\Phi(\alpha, \beta, v) = \begin{cases} \frac{\alpha+\beta}{\|\alpha+\beta\|} & \alpha \neq 0 \text{ and } \beta \neq 0 \\ \Pi(v) & \text{otherwise.} \end{cases} \quad (2)$$

Note that if α and β are taken as in **Step 4)**, then $\alpha + \beta = x^{*3} - x^{*1}$. When $\alpha = 0$ or $\beta = 0$, the direction is updated by the function $\Pi : \mathbb{R}^2 \to \mathbb{R}^2$, which is designed to force the algorithm to explore a large set of directions. Precisely, Π is such that $\forall u^0 \in \mathbb{S}^1$, the set $\{u \in \mathbb{S}^1 : u = \Pi^m(u^0),\ m \in \mathbb{Z}_{>0}\}$ is dense in $\mathbb{S}^1$. One could design Π to rotate the vector by a rational angle (in radians). The simulations in the sequel use this implementation.

Fig. 1 illustrates a typical execution of *RSP* on a quadratic function with an initial position $x^0 \in \mathbb{R}^2$ and initial search direction $v \in \mathbb{S}^1$. When $\varphi \in \mathcal{Q}_2$, each line minimization in a direction v projects the algorithm position onto a line, l_v, that passes through the minimizer. After the first three line minimizations, *RSP* calculates $v^+ = \frac{x^{*3}-x^{*1}}{\|x^{*3}-x^{*1}\|}$ which points towards x^* from x^{*3} and is *conjugate* to v.

Fig. 1 also suggests the result stated in the following Theorem. The proof of this result relies on results relating the conjugacy of vectors and quadratic functions in [12].

Theorem 3.1: Given $\varphi \in \mathcal{Q}_2$, for any initial point $x^0 \in \mathbb{R}^2$ and for any initial direction $v \in \mathbb{S}^1$, the RSP algorithm converges to x^*, the global minimum of φ, in no more than four line minimizations.

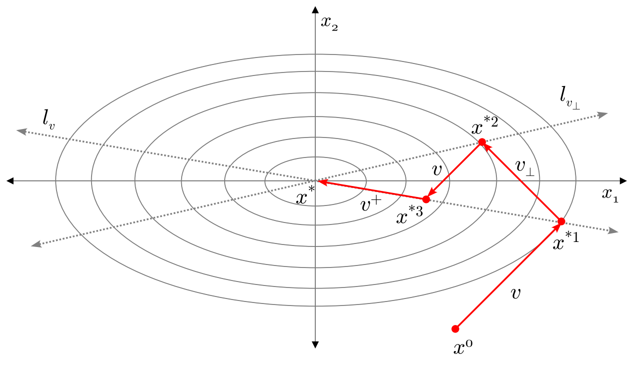


Fig. 1. A typical execution of *RSP* on a quadratic function. Convergence to x^* is achieved after four line minimizations. The ellipses (gray) denote level sets of the quadratic function. The lines l_v and $l_{v_\perp}$ denote the set of points which are minima in the direction of v and $v_\perp$, respectively.

B. RSP with a Practical Line Minimization

The line minimization is the backbone of *RSP*; however, calculating the exact minimum along a line may be impractical or even impossible without more information about the function. This has motivated several methods for conducting a practical line minimization. We propose a method which takes fixed steps of length d in the current search direction until the possibility of a sufficient decrease is exhausted.

Many line minimization algorithms employ the idea of computing a *bracket* (see [9]), which is an interval known to contain a local minimum; then, a line minimization consists of locating a bracket, then finding the minimum of φ within the bracket. A general algorithm for conducting a line minimization this way is pictured in Fig. 2.

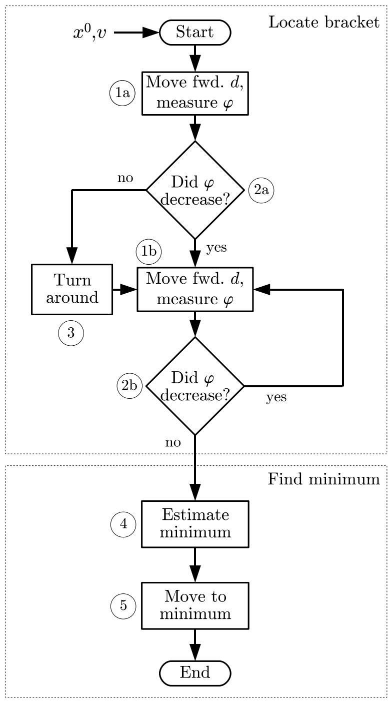


Fig. 2. A flowchart for conducting a line minimization from x_0 in the direction v. A bracket is located when both positive and negative movements of d from a point do not yield a sufficient decrease in φ along v.

We implement a practical line minimization which we call a *discretized line minimization with σ threshold* (a (d,σ)-DLM). During a (d,σ)-DLM, φ is sampled at every step. After each step (blocks 1a & 1b), the question "Did φ decrease?," (blocks 2a & 2b) is answered by comparing the samples. If the samples of φ show a decrease of at least σ, then the answer is "yes." When the answer is "no," the (d,σ)-DLM terminates its search and estimates the minimum. The (d,σ)-DLM simply picks the last measurement point of φ that yielded a decrease of σ as the minimum. Fig. 3 illustrates this entire process. Here, the vehicle first detects an increase in φ and must search in the opposite direction. A bracket is eventually located when an increase in φ over a step is detected directly after a decrease in φ in the previous step.

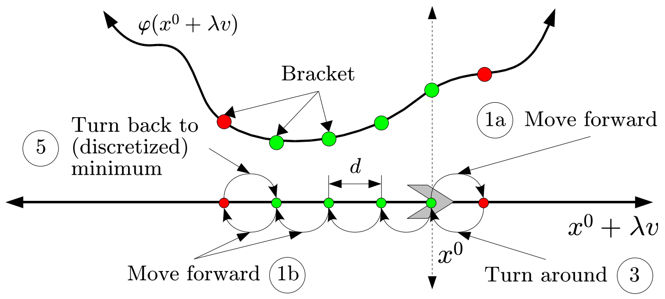


Fig. 3. A (d,σ)-DLM illustration. Green (lighter) dots represent measurements of φ where the vehicle has decided to proceed. The red (darker) dots represent the points where the measurement did not indicate a sufficient decrease.

The result of a (d,σ)-DLM in the direction of v from x^0 is a point $\hat{x}^*$ that satisfies $\varphi(\hat{x}^*) \le \varphi(\hat{x}^* \pm dv) + \sigma$. We denote the set of these points as the (d,σ)-*turning tube* for the direction v,

$$T_\varphi(d,\sigma,v) = \{x \in \mathbb{R}^2 : \varphi(x) \le \varphi(x \pm dv) + \sigma\}. \tag{3}$$

To facilitate the analysis of the hybrid controller, we first analyze a constrained discrete-time system corresponding to *RSP* with a (d,σ)-DLM. We define the states $(x,\xi) \in \mathbb{R}^2 \times \mathbb{R}^\kappa$, where $\xi = (\lambda_1,\lambda_2,v,p,k) \in \Upsilon \subset \mathbb{R}^\kappa$ and $\Upsilon = \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{S}^1 \times \{-1,1\} \times \{0,1,2\}$. The states included in ξ are similar to those in [11]: λ_1,λ_2 record the vectors traveled during the current and previous line minimizations, respectively, v stores the current search direction, p is a logic variable that defines the orientation along the current search direction, and k is a logic variable defining the state of *RSP* . Considering only trajectories which start from $B := \mathbb{R}^2 \times \Upsilon$ (this set is also invariant for the following system). We define the following sets and functions,

$$\begin{aligned}
\overline{D} &= \{(x,\xi) \in B : \varphi(x+dv) \le \varphi(x) - \sigma\}\\
\underline{D} &= \{(x,\xi) \in B : \varphi(x+dv) \ge \varphi(x) - \sigma\}\\
\overline{g} &= \begin{bmatrix}(\lambda_1+dv)^T & \lambda_2^T & v^T & p & k\end{bmatrix}^T\\
D_1 &= \{(x,\xi) \in B : p = 1 \text{ and } \|\lambda_1\|_2 \le d\}\\
g_1 &= \begin{bmatrix}\lambda_1^T & \lambda_2^T & -v^T & -p & k\end{bmatrix}^T\\
D_2 &= \{(x,\xi) \in B : (p = 1 \text{ and } \|\lambda_1\|_2 \ge d) \text{ or } p = -1\}\\
g_2 &= \begin{bmatrix}0 & \lambda_1^T & \mathcal{V}(\xi) & 1 & (k+1) \bmod 3\end{bmatrix}^T,
\end{aligned}$$

where

$$\mathcal{V}(\xi) = \begin{cases}\mathcal{R}_{\pi/2}v & k=0\\ \mathcal{R}_{-\pi/2}v & k=1\\ \Phi(\lambda_1,\lambda_2,v) & k=2\end{cases}$$

and $\bmod$ denotes the *modulo* operator. Then, the constrained discrete-time system is given as

$$\left.\begin{aligned}
x^+ &= \begin{cases} x+dv & (x,\xi)\in\overline{D}\\ x & (x,\xi)\in\underline{D}\end{cases}\\
\xi^+ &= \begin{cases} \overline{g}(\xi) & (x,\xi)\in\overline{D}\\ g_i(\xi) & (x,\xi)\in\underline{D}\cap D_i,\ i=1,2\end{cases}
\end{aligned}\right\} \quad (x,\xi)\in B. \tag{4}$$

Remark 3.2: The system above can be written to satisfy certain regularity properties for discrete-time nonlinear systems. This involves explicitly defining the maps for (x^+,ξ^+) as set-valued at points of discontinuity. For example, take $x \in \overline{D}\cap\underline{D}$, then, $x^+ \in \{x+dv, x\}$.

We define $\mathcal{A} = \{0\}\times\{0\}\times\mathbb{S}^1\times\{-1,1\}\times\{0,1,2\} \subset B$, and the Lyapunov function $V(x,\xi) = \varphi(x) - \varphi(x^*)$. With these, one can use existing Lyapunov stability theory, invariance principles, and notions of detectability for discrete-time nonlinear systems to prove the following results. Due to space constraints, the proofs will not be reported here.

Theorem 3.3: For the system (4)*, the set $\mathcal{A}' = \{x^*\}\times\mathcal{A}$ is stable. Moreover, (x,ξ) converges to $R_\varphi(d,\sigma)\times\mathcal{A}$, where*

$$R_\varphi(d,\sigma) = \bigcap_{v\in\mathbb{S}^1} T_\varphi(d,\sigma,v). \tag{5}$$

Theorem 3.3 uses the construction of Φ and Π to arrive at the result through invariance principles. Using the mean value theorem, one can also assert the following result.

Lemma 3.4: For every $\epsilon > 0$ there exist $d > 0$ and $\sigma > 0$ such that for all $x \in R_\varphi(d,\sigma)$, $\|\nabla\varphi(x)\| < \epsilon$.

The following theorem is a result of Lemma 3.4 and the assumptions of continuous differentiability of φ.

Lemma 3.5: Let $c \in \operatorname{rge}\varphi$. Then, for every $\epsilon > 0$, there exists $d > 0$ and $\sigma > 0$ such that $R_\varphi(d,\sigma)\cap L_\varphi(c) \subset x^* + \epsilon\mathbb{B}$.

Then, from Theorem 3.3 and Lemma 3.5, we can assert a semi-global, practical convergence result.

Theorem 3.6: Let $M \subset B$ be compact and let $\epsilon > 0$. Then, there exist $d > 0$ and $\sigma > 0$ such that for all solutions of (4) *starting from M, (x,ξ) converges to $(x^* + \epsilon\mathbb{B})\times\mathcal{A}$.*

C. Steering the Dubins Vehicle

We solve the open-loop control problem for (1):

> **Given** (x_i,θ_i) and (x_f,θ_f), compute a control law $\omega(t)$ which drives the state of (1) from (x_i,θ_i) to (x_f,θ_f) in finite time.

We parameterize a "bang-bang" control law:

$$\omega(t) = \begin{cases} 0 & t\in[0,t_1)\\ \bar{\omega} & t\in[t_1,t_2)\\ 0 & t\in[t_2,t_3)\\ \bar{\omega} & t\in[t_3,t_4],\end{cases} \tag{6}$$

where $|\bar{\omega}| = \gamma/\rho$. In this way, the vehicle is made to go straight (S) when $\omega(t) = 0$. When $\omega(t) = \bar{\omega}$, the vehicle follows a curve (C) of radius ρ. Note that we allow only right-hand turns or only left-hand turns. This is pictured in Fig. 4, where the S_i's denote the straight paths traveled by the vehicle and the C_i's denote the curves traversed. Following the notation in [7], a path that is made from concatenated C or S segments is denoted as a string of C's and S's. For example, a CSC path means that the vehicle path from some point x_i to x_f is made up of 3 distinct segments, a curve, a straight segment, and a curve, in that order.

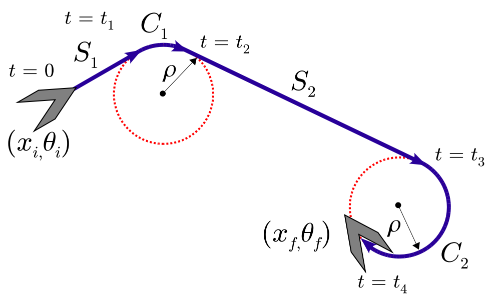


Fig. 4. Steering the Dubins vehicle to an arbitrary state. The vehicle starts at an initial state (x_i, θ_i) and is driven to a final state (x_f, θ_f) in finite time along a path consisting of straight line segments (S_i) and curves (C_j) of radius ρ.

To compute the times t_i, we solve a two-point boundary-value problem. After integration, we arrive at the following system of equations:

$$\begin{aligned} \theta_f &= \bar{\omega}(t_2 - t_1 + t_4 - t_3) \\ \tfrac{x_{1f}}{\gamma} - t_1 - \tfrac{\sin(\theta_f)}{\bar{\omega}} &= (t_3 - t_2)\cos(\bar{\omega}(t_2 - t_1)) \\ -\tfrac{x_{2f}}{\gamma} + \tfrac{1}{\bar{\omega}} - \tfrac{\cos(\theta_f)}{\bar{\omega}} &= (t_3 - t_2)\sin(\bar{\omega}(t_2 - t_1)). \end{aligned} \tag{7}$$

Making the substitutions,

$$\begin{aligned} z_1 &= \tfrac{x_{1f}}{\gamma} - t_1 - \tfrac{\sin(\theta_f)}{\bar{\omega}} \qquad & r &= t_3 - t_2 \\ z_2 &= -\tfrac{x_{2f}}{\gamma} + \tfrac{1}{\bar{\omega}} - \tfrac{\cos(\theta_f)}{\bar{\omega}} \qquad & \phi &= \bar{\omega}(t_2 - t_1), \end{aligned} \tag{8}$$

we arrive at a set of equations in polar form,

$$z_1 = r\cos(\phi) \qquad z_2 = r\sin(\phi), \tag{9}$$

which is easily solved for r and ϕ, in terms of t_1. From (7), (8), and (9), we can find t_2, t_3, and t_4 in terms of t_1. We note here that t_1 is free for us to choose and is usually taken to be zero (which corresponds to a CSC path); however, when x_i and x_f are close together compared to the length of the turning radius, ρ, t_1 can be selected to obviate unnecessary turns, instead producing a SCS path (making $t_4 - t_3 = 0$) that is shorter than a CSC path.

While optimal paths have been studied for the Dubins vehicle, our solution above is sub-optimal. As pointed out in [8, Appendix], these control laws are easy to generate and save on computational resources.

While the derivation above solves the open-loop control problem in a general fashion, we will apply it in only specific situations. Following Fig. 2, in order to carry out the algorithm posed in Section III-B, the autonomous vehicle must be able to execute the following maneuvers.

1) (Blocks 1a,b) Drive the vehicle forward to the next measurement;
2) (Block 3) Turn the vehicle around to the previous measurement in the opposite direction;
3) (Block 5) When a bracket is found, turn the vehicle to the estimated minimum, ready to proceed in the next direction (according to the algorithm).

Maneuver 1) is easily conducted by setting $\omega = 0$ for d/γ seconds. Maneuvers 2) and 3) use the derived control law for the initial state $(0, 0)$ and final state $(-d, \theta_f)$, where $\theta_f = \pi$ when 2) is needed. For reasons derived from geometric and trigonometric arguments, we use the derived control law with the following arguments. When $\theta_f \in [-\pi/2, \pi/2]$ and $d < d^* := |\rho(1 - \cos(\theta_f))/\sin(\theta_f)|$, $t_1 = (d^* - d)/\gamma$, otherwise, $t_1 = 0$. The sign of the angular rate, $\bar{\omega}$, is opposite to the sign of θ_f. We denote the control law using these functions and parameters as $\omega^*(t, \theta_f)$. Given θ_f, we denote the time required by ω^* to reach $(-d, \theta_f)$ (time t_4 in (6)) as $T_\omega(\theta_f)$.

With these values, it can be shown that the following holds. It states that during open-loop turns, which take a finite amount of time, the x state remains in a compact set.

Lemma 3.7: There exists $T^* > 0$ *and* $K^* > 0$ *such that, for every* $\theta_f \in [0, 2\pi]$, *the control law* $\omega^*(t, \theta_f)$ *satisfies* $T_\omega(\theta_f) \le T^*$ *and* $|x(t) - x_f| < K^* \le d + \gamma T^*$ *for all* $t \in [0, T_\omega(\theta_f)]$.

D. The Hybrid Closed Loop

In this section, we integrate the discrete-time optimization algorithm (4) with the vehicle control strategy presented in Section III-C. The vehicle has a state $(x, \theta) \in \mathbb{R}^2 \times \mathbb{R}$. The controller has the state ξ, as present in (4), and also, $(z, \tau, m) \in \mathbb{R} \times \mathbb{R} \times \{0, 1\}$. The state z stores the previous measurement of $\varphi(x)$, τ is a timer needed for open-loop maneuvers, and m is a logic variable. When $m = 0$, the vehicle is conducting a line minimization (and going straight); when $m = 1$, the vehicle is turning to the previous measurement in a new search direction. We group these states as $\zeta = (x, \theta, \xi, z, \tau, m)$ and define the sets

$$\begin{aligned} B' &= \mathbb{R}^2 \times \mathbb{R} \times \Upsilon \times \mathbb{R} \times [0, T^*] \times \{0, 1\} \\ \overline{D}' &= \{\zeta \in B' : \varphi(x) \le z - \sigma\} \\ \underline{D}' &= \{\zeta \in B' : \varphi(x) \ge z - \sigma\} \\ D_1' &= \{\zeta \in B' : (x, \xi) \in D_1\} \\ D_2' &= \{\zeta \in B' : (x, \xi) \in D_2\} \\ M_0 &= \{\zeta \in B' : m = 0\} \\ M_1 &= \{\zeta \in B' : m = 1\} \\ D &= \{\zeta \in B' : m = 0 \text{ and } \tau = d/\gamma \text{ or} \\ &\qquad m = 1 \text{ and } \tau = T_m(r(\xi))\} \\ C &= \{\zeta \in B' : m = 0 \text{ and } \tau \le d/\gamma \text{ or} \\ &\qquad m = 1 \text{ and } \tau \le T_m(r(\xi))\}. \end{aligned}$$

Then, the hybrid closed-loop system is

$$\left.\begin{aligned}
\dot{x} &= \gamma \begin{bmatrix}\cos(\theta)\\ \sin(\theta)\end{bmatrix}\\
\dot{\theta} &= \omega\\
\dot{\xi} &= 0\\
\dot{z} &= 0\\
\dot{\tau} &= 1\\
\dot{m} &= 0\\
\omega &= \begin{cases}0 & m=0\\ \omega^*(\tau,\Theta_{\mathcal{R}}(\xi)) & m=1\end{cases}
\end{aligned}\right\}\ \zeta\in C$$

$$\left.\begin{aligned}
x^+ &= x\\
\theta^+ &= \theta\\
\xi^+ &= \begin{cases}\overline{g}(\xi) & \zeta\in M_0\cap \overline{D}'\\ \xi & \zeta\in M_0\cap \underline{D}'\\ g_i(\xi) & \zeta\in M_1\cap D_i',\ i=1,2\end{cases}\\
z^+ &= \varphi(x)\\
\tau^+ &= 0\\
m^+ &= \begin{cases}0 & \zeta\in (M_0\cap\overline{D}')\cup M_1\\ 1 & \zeta\in M_0\cap \underline{D}'\end{cases}
\end{aligned}\right\}\ \zeta\in D \tag{10}$$

The function $\Theta_{\mathcal{R}}(\xi)$ calculates the angle between the current search direction, v, and the next search direction, either $-v$ or $\mathcal{V}(\xi)$, depending on ξ. With this implementation, the vehicle samples φ, then drives forward to take the next measurement. If this next measurement is σ less than the previous, the vehicle continues along the current search direction, otherwise, the open-loop control law $\omega^*(\tau,\Theta_{\mathcal{R}}(\xi))$ steers the vehicle back to the previous measurement in the new search direction.

The essential property of this system is that it behaves much like the discrete-time system (4). By examining the solutions of (10) and applying the results of Section III-B, one can assert the following convergence and stability results.

Theorem 3.8: For any initial condition, ζ, of (10)*, solutions remain bounded. Moreover, for every $\epsilon > 0$, there exist $\delta > 0$, $d > 0$, $\sigma > 0$, and initial conditions for (ξ, z, τ, m) such that for every initial condition of (x, θ), with $x \in x^* + \delta\mathbb{B}$, x remains in $x^* + (K^* + \epsilon)\mathbb{B}$.*

Theorem 3.9: Let $M \subset B'$ be compact. Then, for every $\epsilon > 0$, there exist $d > 0$ and $\sigma > 0$ such that for all solutions of (10) *starting from M, every state remains bounded and the x trajectories converge to $x^* + (K^* + \epsilon)\mathbb{B}$.*

Remark 3.10: Theorems 3.9 and 3.8 state that the stability and semi-global practical convergence of the discrete-time system (4) is preserved by the hybrid system (10), modulo the extra maneuvers that the vehicle must take to re-orient itself (this is where K^* manifests itself). It is possible to initialize (ξ, z, τ, m) so that the behavior of the hybrid system does not immediately mirror the behavior of the discrete-time system (4) (although, this discrepancy disappears after a bounded amount of time). The initial conditions in Theorem 3.8 are just those initializing *RSP* to its first step.

Remark 3.11: We consider hybrid systems as described in [10], where the dynamics are given by a flow map F, a flow set C, a jump map G, and a jump set D. The function F governs continuous evolution of the state when $x \in C$ and G governs discrete jumps of the state when $x \in D$. Benefits of this framework are revealed when the data of the hybrid system, (F, G, C, D), satisfy mild regularity properties. In this case, a wide range of robust stability analysis tools including invariance principles are available. The system (10) can be written to satisfy these regularity conditions following Remark 3.2.

IV. Experimental and Simulation Results

To illustrate the convergence results presented in Section III-D, we present data from an experiment with a simulated quadratic potential function. The experiment was conducted with an ActivMedia Robotics Pioneer 2-DXE [1] mobile robot and a Vicon MX motion capture system [2]. The experimental setup, pictured in Fig. 5, allows the mobile robot to query the motion capture system for real-time position and orientation information.

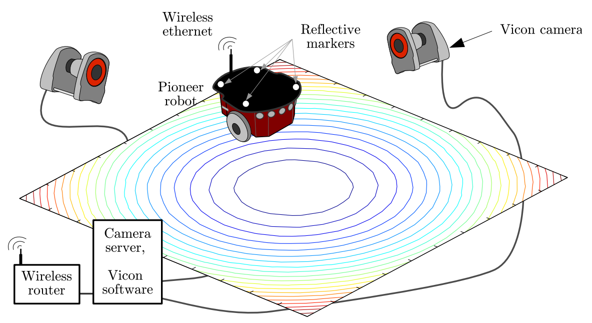


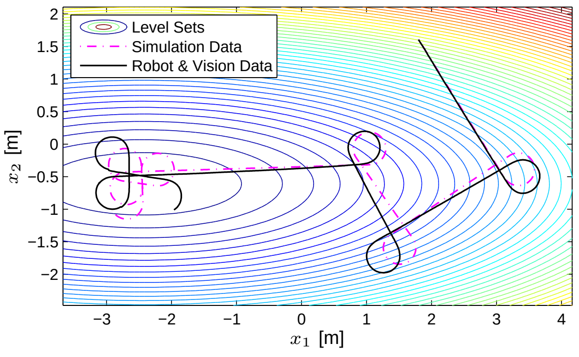
Fig. 5. An experimental setup involving a vision system, mobile robot, and a wireless network. A Vicon camera system is used to measure the position of the robot. Position measurements are sent to the robot and used to simulate the function to be minimized. The hybrid controller on the robot uses the function values to steer the robot towards the minimum.

Using the Real Time Workshop capabilities in MATLAB/Simulink, executable code was compiled for the mobile robot which queried the Vicon MX motion capture system for robot position information, simulated the potential function with the position data, implemented the proposed hybrid controller, and ultimately steered the vehicle. For implementation in a digital computer, the Real Time Workshop discretized the continuous elements of the hybrid controller using a fixed sampling period T_s.

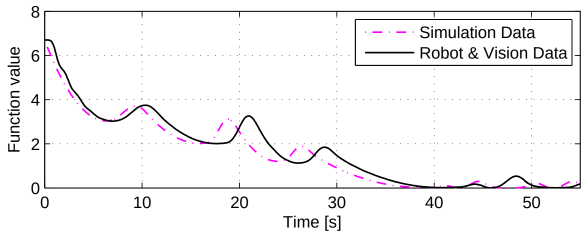
The following experiment was conducted with $\varphi(x) = 0.1(x_1 + 2.44)^2 + (x_2 + 0.61)^2$ and the following parameters in Table I. Results of the experiment are shown in Figures 6(a) & 6(b), along with a computer simulation of the same scenario. No artificial measurement noise was added to $\varphi(x)$, although measurement errors in x are inherent in the camera system. Fig. 6 illustrates the convergence of the algorithm and the closeness of solutions under small perturbations. Differences in the trajectories are mainly attributed

Parameter	Value
$x_1(0)$ [m]	1.80
$x_2(0)$ [m]	1.60
$\theta(0)$ [rad]	-1.01
d [m]	0.1
σ	0.01
T_s [s]	$1/25$
γ [m/s]	0.35
ρ [m]	0.25
$\bar{\omega}$ [rad/s]	1.4

TABLE I
EXPERIMENTAL PARAMETERS



(a) Vehicle trajectories



(b) Function value vs. time

Fig. 6. A comparison of the robot data taken during an experiment with its predicted data from a computer simulation. The robot reaches a neighborhood of the minimum. Behavior of the robot differs slightly from the simulation.

to a delay in the control signal sent to the motors on the robot.

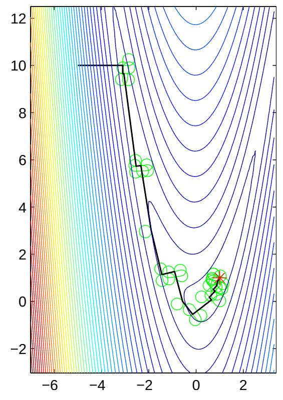
Fig. 7 depicts a simulation of the proposed controller converging to the neighborhood of a non-convex function,

$$\varphi(x) = \sqrt{10(x_2 - x_1^2)^2 + 5(1 - x_1)^2}. \tag{11}$$

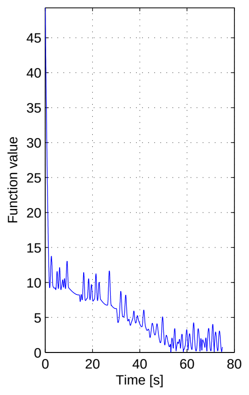
In this simulation, the calculation of conjugate directions aids the vehicle in traversing the narrow valleys of the function.

V. ACKNOWLEDGMENTS

The authors would like to thank Shari Eskenas and Mehmet Usta for their participation in generating the experimental results.



(a) Vehicle trajectory



(b) Function value

Fig. 7. A simulation of the algorithm on the function (11). $d = 0.05$, $\sigma = 0.01$, and $\gamma = 1$. (a) The thicker black line is the path that the algorithm takes. The thinner green line denotes where the vehicle is turning.

REFERENCES

[1] ActivMedia Robotics. http://www.activrobots.com/.

[2] Vicon. http://www.vicon.com/.

[3] R. Bachmayer and N.E. Leonard. Vehicle networks for gradient descent in a sampled environment. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 1, pages 112–117, 2002.

[4] J. Borges de Sousa, K.H. Johansson, J. Silva, and A. Speranzon. A verified hierarchical control architecture for coordinated multi-vehicle operations. *International Journal of Adaptive Control and Signal Processing*, 21:159–188, 2006.

[5] E. Burian, D. Yoerger, A. Bradley, and H. Singh. Gradient search with autonomous underwater vehicles using scalar measurements. In *Proceedings of the 1996 Symposium on Autonomous Underwater Vehicle Technology*, pages 86–98, 1996.

[6] J. Cochran and M. Krstic. Source seeking with a nonholonomic unicycle without position measurements and with tuning of angular velocity - Part I: Stability analysis. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 6009–6016, 2007.

[7] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American J. of Mathematics*, 79(3):497–516, July 1957.

[8] J. A. Farrell, S. Pang, and W. Li. Chemical plume tracing via an autonomous underwater vehicle. *IEEE Journal Of Oceanic Engineering*, 30(2):428–442, April 2005.

[9] R. Fletcher. *Practical methods of optimization; (2nd ed.)*. Wiley-Interscience, New York, NY, USA, 1987.

[10] R. Goebel and A.R. Teel. Solutions to hybrid inclusions via set and graphical convergence with stability theory applications. *Automatica*, 42(4):573–587, April 2006.

[11] C.G. Mayhew, R.G. Sanfelice, and A.R. Teel. Robust source seeking hybrid controllers for autonomous vehicles. In *Proceedings of the 2007 American Control Conference*, pages 1185–1190, 2007.

[12] M.J.D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, 1964.

[13] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, 1960.

[14] C.S. Smith. The automatic computation of maximum likelihood estimates. Technical Report S.C. 846/MR/40, N.C.B., 1961.

[15] C. Zhang, D. Arnold, N. Ghods, A. Siranosian, and M. Krstic. Source seeking with nonholonomic unicycle without position measurement – Part I: Tuning of forward velocity. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006.