A Navigation and Control Strategy for Miniature Legged Robots

Konstantinos Karydis,† Ioannis Poulakakis,‡ and Herbert G. Tanner ‡

***Abstract*—The paper reports on a model-based control strategy for miniature legged robots tasked with navigation in cluttered environments. Our approach uses a new model for crawling locomotion, in order to derive closed-form expressions of state propagation, which then enable the development of a feedback control navigation strategy. The strategy consists of a waypoint tracking controller that steers the system along desired paths, and an outer control loop that updates the reference path to account for uncertainty. This strategy allows for noise-resilient navigation for miniature legged robots, and is experimentally validated on an eight-legged robot that navigates in obstacle-cluttered environments.**

***Index Terms*—Miniature legged robots; Small-scale navigation and control.**

I. Introduction

Advances in our understanding of legged locomotion at small scales [1], [2], as well as the advent of novel manufacturing processes [3], [4] have led to biologically-inspired robot designs with exciting new mobility capabilities. As robot capabilities increase, the introduction and implementation of navigation and control techniques at this scale will expand the capacity of miniature robots to explore, manipulate, and act as remote sensors in a range of useful applications such as search-and-rescue, surveillance, and wildlife monitoring.

The growth, however, in the area of miniature legged robot design and manufacturing [4]–[10], has not been matched by a solid understanding on how control can be realized on these crawlers. With only a few exceptions [11]–[13], such analysis is scarce. This paper partially fills this gap by improving the understanding of how control can be effectively applied at this scale—given the constraints and limitations on payload, energy utilization, and computational power. This in turn is likely to drive design decisions, and possibly feed back to biology by shedding light on the trade-offs in locomotion [14].

Current limitations in manufacturing and instrumentation of miniature robots pose challenges to developing and applying planning, navigation, and control methods in this scale. Size and weight specifications constrain power density, which in turn prohibits extensive reliance on feedback control [5] and restricts power autonomy. One way to mitigate constraints of computational nature is by performing some of the processing off-line [11]. While circumventing the problem of limited on-board computational capacity, this approach shifts some of the challenges to communication, hindering real-time execution. Workarounds exist [13]; after substantial off-line pre-computation, the actual control action can be computed on-board in real time. The latter approach also offers probabilistic performance guarantees, assuming unbounded control effort and no temporal limits on the completion of a navigation task.

Navigation at the miniature scale can be achieved by the hierarchical model-based approach for planning, navigation, and control described in this paper. The approach employs the new Switching Four-Bar Mechanism (SFM) model [15] to create motion planning solutions and generate feedback control loops for navigation at this scale. Motion primitives are constructed and integrated into a Rapidly-exploring Random Tree (RRT) solver [16, Section 14] to generate platform-compatible paths in environments populated with obstacles. A low-level, SFM model-based controller reduces the tracking error, while an outer control loop updates the reference path in a receding-horizon fashion to account for process noise. The strategy is implemented and tested experimentally on the miniature legged robot OctoRoACH [6] (Fig. 1). Experimental results suggest that with limited off-line computing, the reported approach enables the robot to perform real-time navigation tasks utilizing quantized and limited actuation (cf. [11], [13]). The SFM model links high-level navigation objectives to physically implementable control strategies.

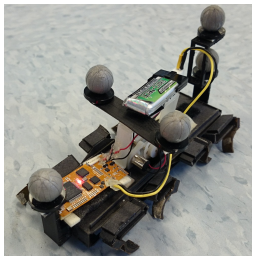


Fig. 1. The miniature eight-legged OctoRoACH robot [6].

The reason for utilizing SFM is related to its capacity to capture salient features of robot behavior *and* enable real-time optimization with limited computational resources. Compared to related bio-inspired dynamic models proposed to capture three-dimensional motion (such as roll oscillations [17]), or the intricacies of leg-ground contact (such as sliding [18]), the SFM is amenable to motion planning and feedback control due to its simple structure that offers closed-form expressions that predict motion behavior. While other lower-dimensional kinematic abstractions of mobile robots often employed in motion planning and control—like the unicycle—are also relevant, such models offer little evidence of better performance; instead, there is evidence of over-generalization to behaviors not attainable by small legged robots such as the OctoRoACH (like turning in place, or going backward). The SFM, in contrast, is comparatively *conservative*: it does not endow the system with more kinematic behaviors than the physical system actually has.

This work extends previous results which introduced SFM [15], and tested in simulation its efficacy in RRT planning [19]. Specifically, this paper utilizes closed-form expressions of state propagation in SFM, to develop a low-level waypoint tracking controller. The scheme is wrapped around an outer control loop that addresses uncertainty, and the whole architecture is experimentally evaluated on the OctoRoACH.

II. Preliminaries

A. The Robot

The OctoRoACH [6] (Fig. 1) is a miniature eight-legged robot that uses two actuators for differential steering; K_L and K_R are motor

This work is supported in part by NSF under grant IIS-1350721, and by ARL MAST CTA # W911NF-08-2-0004.

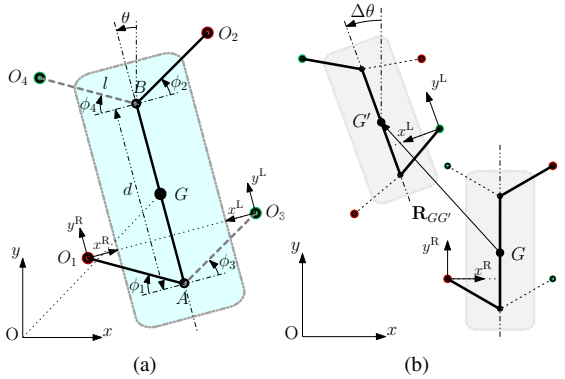
† GRASP Lab, Department of Mechanical Engineering and Applied Mechanics, University of Pennsylvania, USA. Email: kkarydis@seas.upenn.edu

‡ Department of Mechanical Engineering, University of Delaware, USA. Email: {poulakas, btanner}@udel.edu

gains that regulate left and right leg velocities, respectively, which are measured by back-EMF 12-bit A/D units.[1] Changing the desired leg velocities results in paths of varied curvature: (i) straight-line paths (SL) when $K_L = K_R$, (ii) clockwise turns (CW) when $K_L > K_R$, and (iii) counter-clockwise turns (CCW) when $K_L < K_R$. These three types of paths are used in the design of the family of *motion primitives* considered in this analysis. Table I lists the primitives at hand, and gives the resulting motor velocities realizing them.

B. The Switching Four-Bar Mechanism Model

A schematic of the Switching Four-Bar Mechanism (SFM) model [15] is depicted in Fig. 2(a). The SFM consists of a rigid torso and four rigid legs organized in two pairs, the right $\{AO_1, BO_2\}$ and the left $\{AO_3, BO_4\}$. The two pairs are active in turns, while the tips of the active legs form hinged joints with the ground—see Fig. 2(b)—during each step. The motion of the model is parameterized by the leg *touchdown* and *liftoff* angles ϕ_i^{td} and ϕ_i^{lo}, respectively (where $i = 1, \ldots, 4$), and is determined by a single degree of freedom. This feature is exploited to develop analytic expressions for state propagation, and a closed-loop tracking controller.



(a) (b)

Fig. 2. (a) The SFM model. Two pairs of rigid legs become active in turns, forming two fourbar linkages, $\{O_1A, AB, BO_2\}$ and $\{O_3A, AB, BO_4\}$. d is the distance between the two hip-point joints A and B, l denotes the leg length, and G is its geometric center. (b) Pictorial representation of state propagation—the active pair is marked with thick solid lines.

TABLE I
MOTION PRIMITIVES AND IDENTIFIED MODEL PARAMETERS

Family Type	Description	ID	Motor Inputs (K_L, K_R)	$\bar{\phi}^{\text{td}}$ [deg]	$\bar{\phi}^{\text{lo}}$ [deg]
SL	Straight Line	1*	$(60, 60)$	89.84	34.66
		2	$(40, 40)$	89.84	43.55
		3	$(80, 80)$	89.84	24.64
CW	Clockwise	4*	$(60, 20)$	51.11	27.27
		5	$(80, 50)$	51.11	12.03
		6	$(100, 20)$	51.11	20.63
		7	$(100, 60)$	51.11	-2.29
CCW	Counter Clockwise	8*	$(20, 60)$	24.64	1.38
		9	$(40, 80)$	24.64	-11.46
		10	$(20, 100)$	24.64	-1.15
		11	$(60, 100)$	24.64	-19.48

[1]See [6] for details on the design and control architecture of the robot.

C. Parameter Identification

The primitives of Table I are constructed by collecting open-loop planar position and orientation measurements through a VICON motion capture system at 30 Hz. For each primitive, 30 paths are collected. All trials are conducted on a rubber floor mat surface and last 3 sec.[2] The geometric model parameters are specified at $d = 13$ cm (equal to the length of the actual platform), and $l = 3$ cm (the robot's half-width), and the number of model steps for each primitive[3] is set at $N = 10$ (i.e. 5 strides). Straight-line motion is generated by activating both left and right pairs of legs, with the same touchdown and liftoff configurations: $\phi_1^{\text{td}} = \phi_2^{\text{td}} = \phi_3^{\text{td}} = \phi_4^{\text{td}} = \bar{\phi}^{\text{td}}$ and $\phi_1^{\text{lo}} = \phi_2^{\text{lo}} = \phi_3^{\text{lo}} = \phi_4^{\text{lo}} = \bar{\phi}^{\text{lo}}$. Turns are generated as a variation of the above gait where only one pair is active during each stride. For example, clockwise turning is realized by a "degenerate" stride in which the left pair is active, and no motion occurs during the right pair support (i.e., $\phi_1^{\text{td}} = \phi_2^{\text{td}} = \phi_1^{\text{lo}} = \phi_2^{\text{lo}} = 0$). Similarly, counter-clockwise turns happen by right pair activation only, with no motion during the left pair support: $\phi_3^{\text{td}} = \phi_4^{\text{td}} = \phi_3^{\text{lo}} = \phi_4^{\text{lo}} = 0$. With these conventions in place, a constrained least-squares optimization problem is solved to identify the remaining model parameter values that enable the model to generate paths that best capture the experimental averages.[4] The identified model parameters populate the last two columns of Table I.

Remark 1: Model parameters are not directly linked to the physical platform's actuation characteristics. Instead, Table I provides touchdown and liftoff angles for the SFM, and actuator inputs for the OCTOROACH, that produce *the same spatial displacement* between the model and the robot while executing a given primitive. Essentially, Table I serves as lookup table that links model parameter values to robot actuation inputs (motor velocities). As shown shortly, this is important for model-based feedback control commands to be mapped to robot control inputs in real-time.

III. CLOSED-FORM STATE PROPAGATION FOR THE SFM

The state of the model is $q = (x_G, y_G, \theta)^\intercal \in \mathbb{R}^2 \times \mathbb{S}$, where (x_G, y_G) denotes the position of its geometric center in the global coordinate frame $\{\text{O}\}$, and θ is its heading. Positive changes in the heading represent counter-clockwise rotations. Let $\zeta \in Z$ denote the vector of model parameters, which include the touchdown and liftoff angles of each active pair, and f the model state propagation function over one step. This function takes as input the value of the model's single degree-of-freedom variable ϕ, and the set of constant model parameters ζ, and produces the corresponding spatial $\mathbf{R}_{GG'} = (\Delta x, \Delta y)$, and angular $\Delta\theta$ displacement of the model in the local coordinate frame—see Fig. 2(b). One would then write $f : \left[\phi^{\text{td}}, \phi^{\text{lo}}\right] \times Z \to \mathbb{R}^2 \times \mathbb{S}$, and for the corresponding pair of active legs,

$$(\Delta x, \Delta y, \Delta\theta)^\intercal = f(\phi; \zeta) \ . \tag{1}$$

When the right pair is active, the local coordinate frame is attached to O_1, $\phi = \phi_1 \in \left[\phi_1^{\text{td}}, \phi_1^{\text{lo}}\right]$, $\zeta = \left\{\phi_2^{\text{td}}\right\}$, and $(\Delta x, \Delta y, \Delta\theta)^\intercal =$

[2]The time duration of 3 sec offers a reasonable trade-off between robot path dispersion and path length, which in turn affects the computationally complexity of the problem; see [15] for more details.

[3]This number of model steps is chosen empirically over the course of data collections and model analyses to provide adequate resolution for the touchdown and liftoff configurations to capture 3 sec-long experimental data.

[4]Details on the optimization procedure can be found in [15].

$f_{\mathrm{R}}(\phi_1;\phi_2^{\mathrm{td}})$ with

$$f_{\mathrm{R}}(\phi_1;\phi_2^{\mathrm{td}}) = \begin{bmatrix} r(\phi_1)\sin\big(\omega(\phi_1)-|\phi_1-\phi_1^{\mathrm{td}}+\chi(\phi_1,\phi_2^{\mathrm{td}})|\big)-r(\phi_1^{\mathrm{td}})\sin\omega(\phi_1^{\mathrm{td}}) \\ r(\phi_1)\cos\big(\omega(\phi_1)-|\phi_1-\phi_1^{\mathrm{td}}+\chi(\phi_1,\phi_2^{\mathrm{td}})|\big)-r(\phi_1^{\mathrm{td}})\cos\omega(\phi_1^{\mathrm{td}}) \\ \phi_1-\phi_1^{\mathrm{td}}+\chi(\phi_1,\phi_2^{\mathrm{td}}) \end{bmatrix}. \quad (2)$$

Similarly, when the left pair is active, the local coordinate frame is attached at O_3, $\phi = \phi_3 \in [\phi_3^{\mathrm{td}}, \phi_3^{\mathrm{lo}}]$, $\zeta = \{\phi_4^{\mathrm{td}}\}$, and $(\Delta x, \Delta y, \Delta\theta)^\intercal = f_{\mathrm{L}}(\phi_3;\phi_4^{\mathrm{td}})$ with

$$f_{\mathrm{L}}(\phi_3;\phi_4^{\mathrm{td}}) = \begin{bmatrix} -\big(r(\phi_3)\sin\big(\omega(\phi_3)-|\phi_3-\phi_3^{\mathrm{td}}+\chi(\phi_3,\phi_4^{\mathrm{td}})|\big)-r(\phi_3^{\mathrm{td}})\sin\omega(\phi_3^{\mathrm{td}})\big) \\ r(\phi_3)\cos\big(\omega(\phi_3)-|\phi_3-\phi_3^{\mathrm{td}}+\chi(\phi_3,\phi_4^{\mathrm{td}})|\big)-r(\phi_3^{\mathrm{td}})\cos\omega(\phi_3^{\mathrm{td}}) \\ -\big(\phi_3-\phi_3^{\mathrm{td}}+\chi(\phi_3,\phi_4^{\mathrm{td}})\big) \end{bmatrix}. \quad (3)$$

With reference to Fig. 2, and for the right pair, let the quantities $r(\phi_1)$ and $r(\phi_1^{\mathrm{td}})$ represent the lengths of the vectors connecting the tip O_1 to the geometric center of the model at liftoff and touchdown, respectively. The angles $\omega(\phi_1)$ and $\omega(\phi_1^{\mathrm{td}})$ are formed between the above vectors and the model's torso, while angle $\chi(\phi_1,\phi_2^{\mathrm{td}})$ captures the range of rotation of leg $\{O_1A\}$ about O_1, relative to the body frame anchored at O_1, and depends on the touchdown angle ϕ_2^{td}. The same notation is followed for the left pair as well. See [20, Appendix B] for the closed-form expressions of $r(\cdot)$, $\omega(\cdot)$, and $\chi(\cdot)$.

Remark 2: Note that (2) and (3) have the exact same structure, and differ only in the signs of Δx and $\Delta\theta$. This is because of SFM being symmetric about its longitudinal axis.

Letting the right pair go first, the model is initiated at $q^{\mathrm{R}^-} \triangleq (x_G, y_G, \theta)^\intercal$. The touchdown configuration $(\phi_1^{\mathrm{td}}, \phi_2^{\mathrm{td}})$ defines the position of the points O_1 and O_2, which remain fixed for the duration of the step. The model state is propagated in the local frame according to (2) until ϕ_1 reaches its liftoff configuration, ϕ_1^{lo}. The model state at the liftoff configuration—expressed in the local frame—is $(\Delta x, \Delta y, \Delta\theta)^\intercal = f_{\mathrm{R}}(\phi_1^{\mathrm{lo}};\phi_1^{\mathrm{td}},\phi_2^{\mathrm{td}})$, and is then mapped back to the global frame through a homogeneous transformation, yielding q^{R^+}. The state of the model at the end of the right step initiates the left step, with $q^{\mathrm{R}^+} = q^{\mathrm{L}^-}$. From $(\phi_3^{\mathrm{td}}, \phi_4^{\mathrm{td}})$ one finds O_3 and O_4, solves (3) until $\phi_3 = \phi_3^{\mathrm{lo}}$, and maps the (local) model state to q^{L^+}. Then, q^{L^+} initiates the next (right) pair and the cycle continues. The closed-form state propagation equations (2), (3) support the development of a step-by-step state feedback controller that selects liftoff configurations that bring the state of the model closer to a desired one. Section IV that follows shows how the implementation of this strategy gives rise to a waypoint tracking controller that is subsequently applied to the miniature legged robot OctoRoach [6].

IV. Hierarchical Control and Navigation

The reported navigation strategy consists of three distinct phases: first, the motion primitives of Table I are used by a sampling-based motion planner to generate reference paths that link origin and destination points. Then, a (local) low-level controller, exploiting the structure of SFM, is used for tracking waypoints along reference paths. Experimentation with the OctoRoach indicates that the effect of the ground interaction uncertainty on this particular platform is too large for the step-by-step local controller. Thus, the third layer of the hierarchy introduces an *outer control-loop* that updates the reference path while the low-level controller is active.

A. Path Planning

In the first phase, the time-parameterized motion primitives are used in a temporal sequence to generate a collision-free reference path between an initial and desired state. We employ a *Rapidly-exploring Random Tree* [16, Section 14] planner that uses the primitives to generate new vertices.

To limit the method's computational requirements, only three motion primitives—marked with an asterisk in Table I—are used in this phase. Figure 3 shows the constructed reference path from the initial state $q_s = (20, 20, -90)^\intercal$ [cm, cm, deg], to the goal centered on $q_d = (210, 210, -90)^\intercal$ [cm, cm, deg]. Due to the discretization imposed by the primitives, reaching exactly q_d is unlikely, thus acceptable paths are allowed to end within a radius of 15 cm around (x_d, y_d) with an orientation in the range $[-45°, 45°]$.

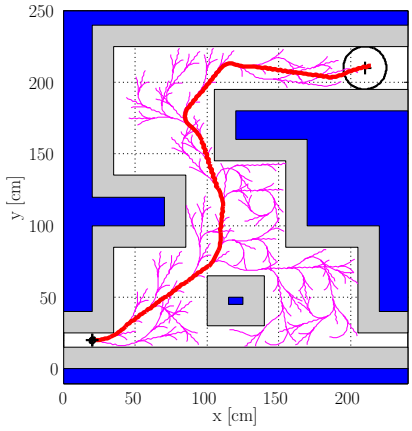


Fig. 3. The RRT is outlined in thin curves, while the desired path is marked with a thick curve. Obstacles (shown in dark shading) are artificially augmented (lightly shaded regions) to account for platform volume.

Significant levels of noise, evident in the experimental results of Section V, do not allow the sequence of motion primitives to reach the goal state when executed in an open-loop fashion; the control loop needs to close.

B. Path Tracking Control

The analytic expressions of Section III facilitate the design of a fast (see Section V-C) closed-loop controller tasked with minimizing the tracking error. With reference to Fig. 4, in every step the controller:

1. extracts from the reference path the desired state at the beginning of the current $q'(m)$ and next $q'(m+1)$ steps;
2. uses the SFM to predict the state $q(m+1)$ at the next step, starting with the actual state $q(m)$ at the current step, under all pairs of touchdown and liftoff angles given in Table I;
3. selects the combination that minimizes the tracking error based on a $\mathsf{SE}(2)$ metric;
4. applies the selected settings to update the control input on the robot (i.e. motor velocities) through Table I.

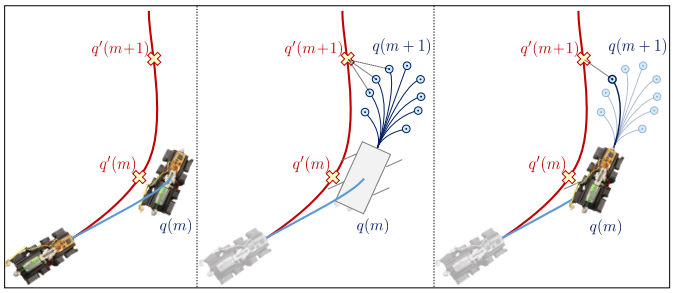


Fig. 4. Pictorial representation of the tracking controller's function. Primed and non-primed quantities denote desired and actual states at the beginning of step m and $m+1$, respectively.

If N are the steps the model takes in each primitive, and K is the number of primitives that compose the reference path, then the latter will consist of $N \cdot K$ steps. For $k \in \{1, \ldots, K\}$, define the mapping $\pi_k : \{N(k-1)+1, Nk\} \rightarrow \mathbb{R}^2 \times \mathbb{S}$; $\pi_k[m]$ produces the desired state for the system (i.e. a waypoint) at the beginning of step m, expressed in the global frame. Then, evaluating π_k for $m \in \{N(k-1)+1, Nk\}$ gives a sequence of reference waypoints for one primitive. Concatenation of these sequences produces the (discrete) reference $\Pi \triangleq \pi_1 \pi_2 \cdots \pi_K : \{1, NK\} \rightarrow \mathbb{R}^2 \times \mathbb{S}$. Now let $\Pi[m] = \left(x'[m], y'[m], \theta'[m]\right)^\intercal$ denote a waypoint along the reference path, and $q[m] = \left(x_G[m], y_G[m], \theta[m]\right)^\intercal$ the actual state at step m. Let $\mathrm{T}(q[m])$ be the homogeneous transformation on $\mathsf{SE}(2)$ that maps the coordinates of the model state from the local frame to the global. For a parameter pair $\left(\bar{\phi}^{\mathrm{td}}, \bar{\phi}^{\mathrm{lo}}\right)$ drawn from Table I, the model-predicted state of the system at the beginning of the next step is

$$\left(x_G[m+1], y_G[m+1], \theta[m+1]\right)^\intercal = \mathrm{T}(q[m])\left(\Delta x, \Delta y, \Delta\theta\right)^\intercal \ . \quad (4)$$

Displacements $\Delta x, \Delta y$, and $\Delta\theta$ are determined by $\left(\bar{\phi}^{\mathrm{td}}, \bar{\phi}^{\mathrm{lo}}\right)$, and are given by the expressions in (2) or (3) depending on the active pair (i.e. right or left, respectively).

If $\left(x'[m+1], y'[m+1], \theta'[m+1]\right)^\intercal$ is the desired state at the beginning of step $m+1$ based on the reference path, the predicted path error at the beginning of step $m+1$ is

$$\begin{aligned} \delta x &= x_G[m+1] - x'[m+1] \\ \delta y &= y_G[m+1] - y'[m+1] \\ \delta\theta &= \ \ \theta[m+1] - \theta'[m+1] \ , \end{aligned}$$

and depends on $\left(\bar{\phi}^{\mathrm{td}}, \bar{\phi}^{\mathrm{lo}}\right)$. If an $\mathsf{SO}(2)$ metric [16, Section 5.1.2] $\rho_\theta(\theta, \theta') \triangleq \sqrt{\left(\cos(\theta)-\cos(\theta')\right)^2 + \left(\sin(\theta)-\sin(\theta')\right)^2}$ is selected, along with the Euclidean metric on $\mathbb{R}^2$, $\rho_{\mathbf{x}}(\mathbf{x}, \mathbf{x}') \triangleq \sqrt{\delta x^2 + \delta y^2}$, then a controller can be devised to pick $\left(\bar{\phi}^{\mathrm{td}}, \bar{\phi}^{\mathrm{lo}}\right)$ from Table I in order to minimize the metric

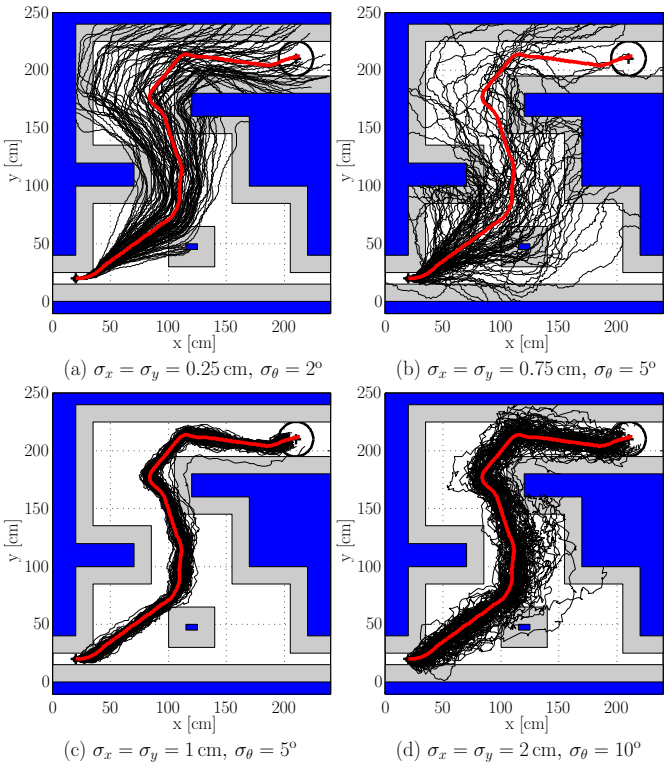
$$\rho(q, q') = \sqrt{\left[\rho_\theta(\theta, \theta')\right]^2 + \alpha_{cl}\left[\rho_{\mathbf{x}}(\mathbf{x}, \mathbf{x}')\right]^2} \ . \quad (5)$$

Coefficient α_{cl} is a relative weight between heading and planar distance discrepancies. Its value here is chosen empirically; it was found that $\alpha_{cl} = 0.015$ offers a good trade-off in penalizing errors in heading and planar distance.

Figure 5 depicts both the open– and closed-loop response of the system in simulation for the case study of Fig. 3, for varying degrees of infused uncertainty, and with noise generated according to a zero-mean normal distribution. In each case we simulate the system 100 times. Figures 5(a)-(b) depict the open-loop response of the system under motion perturbations drawn from zero-mean normal distributions with different variances. Figures 5(c)-(d) show the response of the system for larger motion perturbations when the path tracking control loop is closed. As expected, closing a local path tracking loop drastically improves the chances of the system reaching its destination without collisions (Fig. 5(c)). The figures also reveal the robustness limits of the controller: beyond a certain threshold, and with persistent noise excitation, none of the primitives of Table I can close the gap between the current and desired state on the reference path sufficiently fast (Fig. 5(d)). A second control loop, tasked with compensating for that deficiency, is the subject of the next section.

C. Closing an Outer Control Loop

The outer control loop is closed using the reference path generator, to plan paths in a receding-horizon fashion, with a period of $\delta < K$ primitives. Essentially, the system's low-level controller steers the system as close to the initial path as possible for δ primitives, and then the reference path is recomputed. The cycle is repeated until either the goal is reached, or the system collides with obstacles.



(a) $\sigma_x = \sigma_y = 0.25\,\mathrm{cm}$, $\sigma_\theta = 2^\mathrm{o}$ (b) $\sigma_x = \sigma_y = 0.75\,\mathrm{cm}$, $\sigma_\theta = 5^\mathrm{o}$

(c) $\sigma_x = \sigma_y = 1\,\mathrm{cm}$, $\sigma_\theta = 5^\mathrm{o}$ (d) $\sigma_x = \sigma_y = 2\,\mathrm{cm}$, $\sigma_\theta = 10^\mathrm{o}$

Fig. 5. Simulated response of the system commanded to follow a desired path (thick red curve), when uncertainty affects its state at the end of every step. (a) Open-loop response of the system with low magnitude of infused uncertainty scores a 10% success rate. (b) As the magnitude of the infused uncertainty grows, the success rate reduces to 2%. (c) Closed-loop response of the system using the reported path tracking controller. When the magnitude of the infused uncertainty is low, the controller enables the system to follow the desired path, scoring a 100% success rate. (d) As the magnitude of the infused uncertainty grows, the system may exit the region of attraction of the controller and lose track of the desired path; the success rate reduces to 85%.

V. Experimental Evaluation and Assessment

This section reports on experiments conducted with the OctoRoach navigating in a two-dimensional constrained environment, using the hierarchical control approach of Section IV.

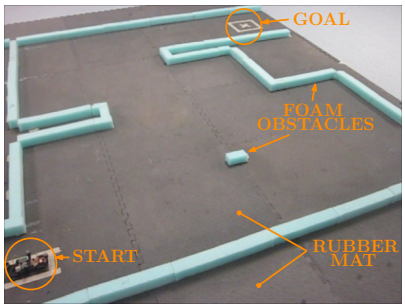


Fig. 6. The physical environment realizing the case study of Fig. 3.

A. Experimental Setup

Figure 6 shows the physical environment of our case study. The robot is placed manually in a designated START position, and the goal is for the platform to reach a rectangular region marked GOAL. A VICON motion capture system provides ground truth position and orientation data. Control computations are performed in Python

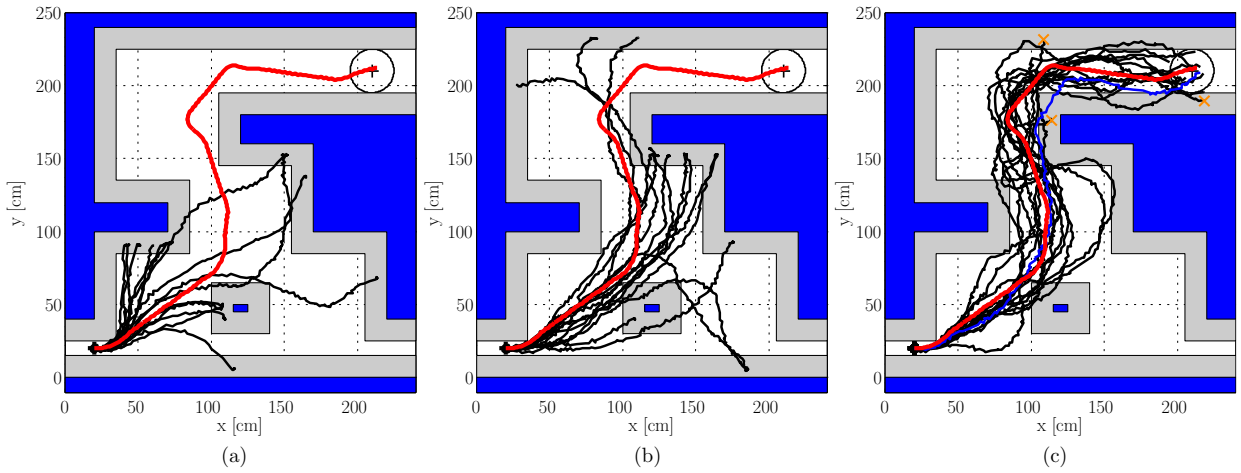


Fig. 7. Experimental results with the OctoRoach. The desired path is marked with a thick curve in all cases. (a) Open-loop implementation of the desired path. (b) Closed-loop response of the system using the reported path tracking controller. (c) The combination of local path tracking control with the prediction phase. The path is refined every 2 primitives using the RRT; Fig. 9 shows this process for one sample path, highlighted here in blue (color version).

running on a host Linux laptop, while reference path generation is done using MATLAB implementations of RRT on the same machine. Software modules interface with the physical platform through ROS. Control inputs (i.e. motor velocities) are sent to the robot at 3.33 Hz; recall a primitive lasts 3 sec and consists of 10 model steps.

B. Results

The desired path shown in Fig. 3 is first executed in open-loop fashion. The results of 15 such trials are shown in Fig. 7(a). The robot is shown to collide with the obstacles soon after it starts navigating. The response of the system under the effect of the step-by-step path tracking controller of Section IV-B, which picks parameter configurations from the available ones shown in Table I and maps this selection to the respective motor velocities, is shown in Fig. 7(b) for a total of 15 trials. In comparison to the open-loop case, longer portions of the reference path are now realized. However, the persistence of process noise in conjunction with the length of the reference path prove to be beyond the capacity of the controller in all fifteen trials.

There is a marked improvement in the rate at which the platform reaches the goal when the outer control loop is also closed. The path is updated every $\delta = 2$ primitives using the RRT planner, and the process repeats until the target is reached, or the robot collides with obstacles. Using this inner/outer closed-loop control scheme leads to an 80% success rate. Figure 7(c) shows the results of applying this control scheme, and Fig. 8 summarizes all path completion rates.

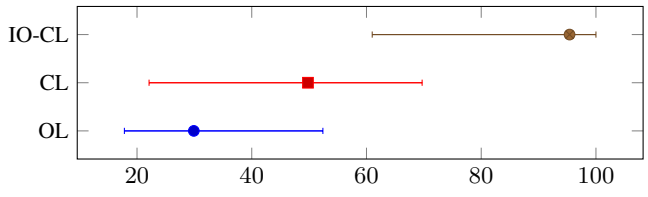


Fig. 8. Percentage of the reference path tracked. Whiskers indicate the minimum and maximum success percentage over all trials and the points mark the average success rate. OL: open-loop. CL: closed-loop (local path tracker). IO-CL: inner/outer closed-loop (local path tracker with replanning).

Figure 9 illustrates in more detail how the inner/outer closed-loop control scheme worked in a specific experimental trial, highlighted in blue (color version) in Fig. 7. The tree that is constructed from a robot state is shown in thin curves, while the thick curve marks the reference path selected by the planner. Each instance in Fig. 9 illustrates a new reference path update, calculated after executing two primitives (approximately 20 robot steps) along the current reference path. On its way to goal, process noise may push the robot in shaded regions (Figs. 9(g)-(h)). In these cases, the RRT is allowed to generate new edges inside the augmented obstacle regions provided that generated paths exit the shaded region after the first primitive.

C. Discussion

The inner/outer closed-loop control scheme proposed here enables the OctoRoach to achieve navigation tasks with high probability of success. However, there are still cases (Fig. 7(c)) in which the robot collides with its environment. These collisions could in principle have been prevented by using a smaller control horizon δ, at the expense of increasing the computational complexity of the navigation task. Selecting the control horizon is an implementation issue in similar receding-horizon schemes; it ultimately relates to the amount of uncertainty and noise the system is subjected to.

Benchmark tests were conducted to identify the computational time required by the proposed approach. It takes on average 1.01 sec for generating a path between the start and goal regions of Fig. 6. This performance was benchmarked by running the RRT planner 1000 times; the standard deviation is 0.29 sec. The local controller that adjusts the robot control inputs based on our closed-form state propagation equations executes fast: it updates the robot control inputs in 5.87×10^{-4} sec on average with standard deviation 3.04×10^{-4} sec. (The sample size in this test was $N = 20745$.) All tests were run in Matlab R2012a on a Windows 10 machine with a 2-core i5 − 33172 CPU @ 1.70GHz and 4 GB DDR3 RAM. It is anticipated that a C / C++ implementation will bring these computation times down.

Remark 3: It is worth mentioning that the employed SFM model can be also used to capture the behavior of other, morphologically distinct miniature legged robots [21]. In this light, the controller can be employed for navigation of other miniature legged robots as well; it is not specialized to the particular robot used in this study.

VI. Conclusions

Navigation at the miniature scale can be achieved by the hierarchical model-based approach for planning, navigation, and control. The approach enables miniature legged robots to navigate in environments

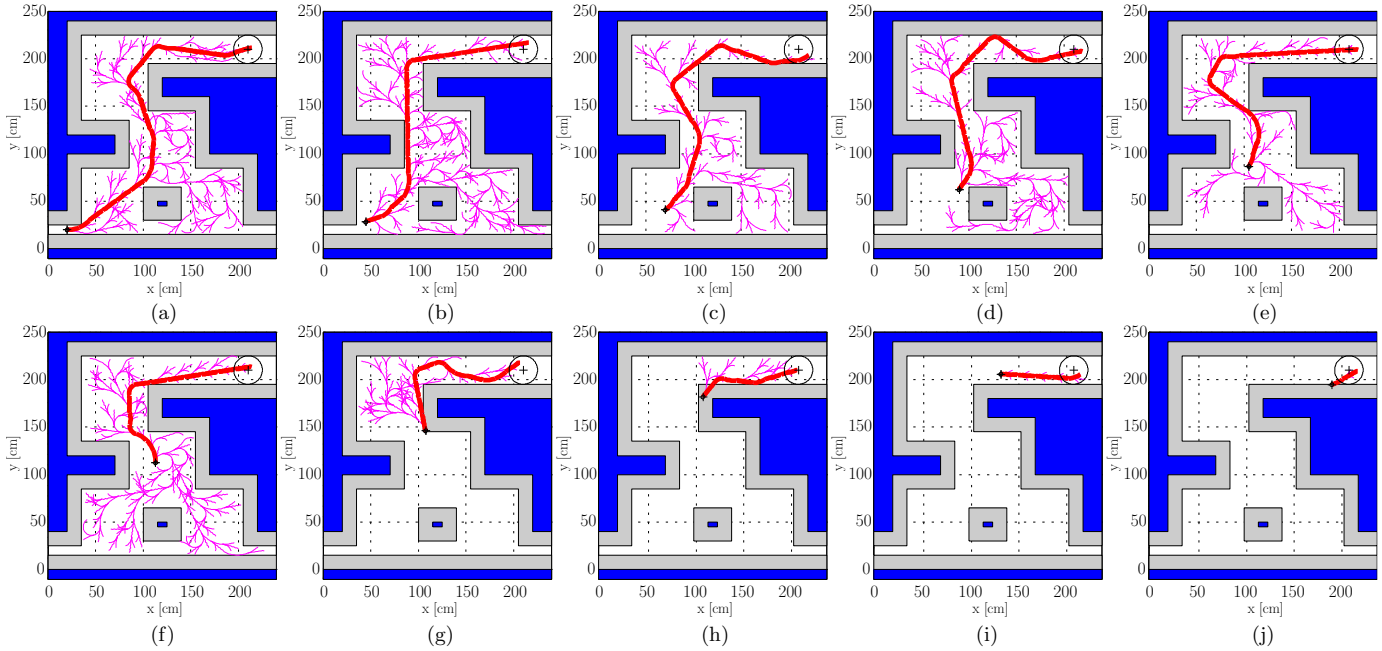


Fig. 9. Illustration of the two-stage path tracking control and prediction scheme when applied to the physical robot. (a) The initial path (thick curve) generated by the RRT (cf. Fig. 3) is updated after the first 2 primitives of the plan have been executed, with the local path tracking controller being active. (b) The updated path (thick curve) is followed for the first 2 primitives, and updated again by the solver. (c)-(j) The process repeats until the desired state is reached.

populated with obstacles, by tracking on-line generated reference paths in the presence of noise. At the core of the approach is the *Switching Four-Bar Mechanism* (SFM), a kinematic model for quasi-static miniature legged locomotion in the horizontal plane. The SFM is used to develop low-level control, and mid-to-high-level navigation strategies, which are implemented and evaluated experimentally on the OctoRoach in navigation tasks when crawling at low speeds. Because of the persistence of process noise in conjunction with the length of the reference path, the low-level controller is supplemented by a second control loop, wrapped around the reference path generator, that works in a receding-horizon fashion.

References

[1] T. Zhang, F. Qian, C. Li, P. Masarati, A. Hoover, P. Birkmeyer, A. Pullin, R. S. Fearing, and D. I. Goldman, "Ground fluidization promotes rapid running of a lightweight robot," *The International Journal of Robotics Research*, vol. 7, no. 32, pp. 859–869, 2013.

[2] C. Li, T. Zhang, and D. I. Goldman, "A terradynamics of legged locomotion on granular media," *Science*, vol. 339, pp. 1408–1412, 2013.

[3] R. Wood, S. Avadhanula, R. Sahai, E. Steltz, and R. S. Fearing, "Microrobot Design Using Fiber Reinforced Composites," *ASME Journal of Mechanical Design*, vol. 130, no. 5, pp. 1–11, 2008.

[4] C. D. Onal, M. T. Tolley, R. J. Wood, and D. Rus, "Origami-inspired printed robots," *IEEE/ASME Transactions on Mechatronics*, vol. PP, no. 99, pp. 1–8, 2014.

[5] A. M. Hoover, S. Burden, X.-Y. Fu, S. Sastry, and R. S. Fearing, "Bio-inspired design and dynamic maneuverability of a minimally actuated six-legged robot," in *IEEE Int. Conf. on Biomedical Robotics and Biomechatronics*, 2010, pp. 869–876.

[6] A. Pullin, N. Kohut, D. Zarrouk, and R. S. Fearing, "Dynamic turning of 13 cm robot comparing tail and differential drive," in *IEEE Int. Conf. on Robotics and Automation*, 2012, pp. 5086–5093.

[7] D. Zarrouk, A. Pullin, N. Kohut, and R. S. Fearing, "STAR - a sprawl tuned autonomous robot," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 20–25.

[8] C. Brown, D. Vogtmann, and S. Bergbreiter, "Efficiency and effectiveness analysis of a new direct drive miniature quadruped robot," in *IEEE Int. Conf. on Robotics and Automation*, 2013, pp. 5631–5637.

[9] A. T. Baisch, O. Ozcan, B. Goldberg, D. Ithier, and R. J. Wood, "High speed locomotion for a quadrupedal microrobot," *The International Journal of Robotics Research*, vol. 33, no. 8, pp. 1063–1082, 2014.

[10] D. W. Haldane, C. S. Casarez, J. T. Karras, J. Lee, C. Li, A. O. Pullin, E. W. Schaler, D. Yun, H. Ota, A. Javey, and R. S. Fearing, "Integrated manufacture of exoskeletons and sensing structures for folded millirobots," *Journal of Mechanisms and Robotics*, vol. 7, pp. 021 011–1–021 011–19, 2015.

[11] A. Mathis, J. Russell, T. Moore, J. Cohen, B. Satterfield, N. Kohut, X.-Y. Fu, and R. S. Fearing, "Autonomous navigation of a 5 gram crawling millirobot in a complex environment," in *Adaptive Mobile Robotics: 15th Int. Conf. on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, 2012, pp. 121–128.

[12] K. Karydis, D. Zarrouk, I. Poulakakis, R. S. Fearing, and H. G. Tanner, "Planning with the STAR(s)," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2014, pp. 3033–3038.

[13] K. S. Shah, C. D. Pahlajani, and H. G. Tanner., "Optimal navigation for vehicles with stochastic dynamics," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 2003–2009, 2015.

[14] N. J. Cowan, M. M. Ankarali, J. P. Dyhr, M. S. Madhav, E. Roth, S. Sefati, S. Sponberg, S. A. Stamper, E. S. Fortune, and T. L. Daniel, "Feedback control as a framework for understanding tradeoffs in biology," *Integrative and Comparative Biology*, vol. 54, no. 2, pp. 223–237, 2014.

[15] K. Karydis, Y. Liu, I. Poulakakis, and H. G. Tanner, "A Template Candidate for Miniature Legged Robots in Quasi-Static Motion," *Autonomous Robots*, vol. 38, no. 2, pp. 193–209, 2015.

[16] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.

[17] D. W. Haldane and R. S. Fearing, "Roll oscillation modulated turning in dynamic millirobots," in *IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 4569–4575.

[18] D. Zarrouk and R. S. Fearing, "Controlled in-plane locomotion of a hexapod using a single actuator," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 157–167, 2015.

[19] K. Karydis, Y. Liu, I. Poulakakis, and H. G. Tanner, "Navigation of miniature legged robots using a new template," in *23rd Mediterranean Conference on Control and Automation*, 2015, pp. 1112–1117.

[20] K. Karydis, "A data-driven hierarchical framework for planning, navigation, and control of uncertain systems: Applications to miniature legged robots," Ph.D. dissertation, University of Delaware, 2015.

[21] K. Karydis, A. Stager, H. G. Tanner, and I. Poulakakis, "Experimental Validation of a Template for Navigation of Miniature Legged Robots," in *Int. Symposium on Experimental Robotics*, 2016, to appear.