2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Automatic Identification of Dynamic Piecewise Affine Models for a Running Robot

Austin D. Buchan*, Duncan W. Haldane*, and Ronald S. Fearing

Abstract— **This paper presents a simple, data-driven technique for identifying models for the dynamics of legged robots. Piecewise Affine (PWA) models are used to approximate the observed nonlinear system dynamics of a hexapedal millirobot. The high dimension of the state space (16) and very large number of state observations (∼100,000) motivated the use of statistical clustering methods to automatically choose the submodel regions. Comparisons of models with 1 to 50 PWA regions are analyzed with respect to state derivative prediction and forward simulation accuracy. Derivative prediction accuracy was shown to reduce average in-axis absolute error by up to 52% compared to a null estimator. Simulation results show tracking of state trajectories over one stride length, and the degradation of simulation prediction is analyzed across model complexity and time horizon. We describe metrics for comparing the performance of different model complexities across one-step and simulation predictions.**

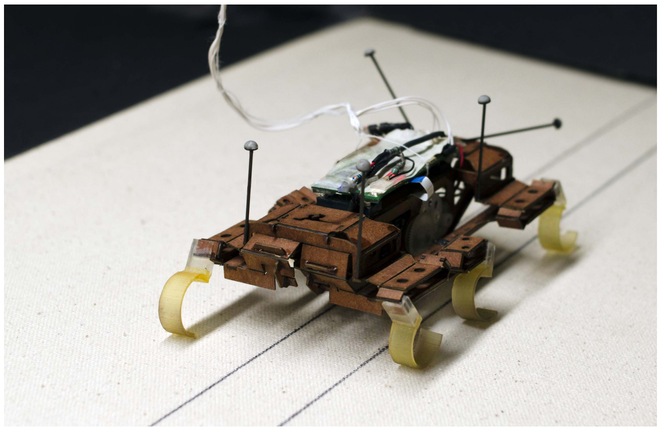


Fig. 1: The VelociRoACH on a treadmill, equipped with tether.

I. Introduction

Biologically inspired millirobots are inexpensive to produce, highly robust, and can exhibit remarkable dynamic performance [6]. Due to their small size, it is important to minimize the number of actuators and the required actuator bandwidth. Therefore, these robots [2][8][6] have been designed to be open-loop stable, allowing them to convert feed-forward motor power into stable and robust locomotion. The ability of these robots to run dynamically has allowed us to observe a number of emergent dynamic maneuvers such as rapid turns, reversals, jumps exceeding body height, flips, and cartwheels. However, we currently have no modeling approaches that are accurate enough to predict these aggressive maneuvers on the time scales significant to our robotic systems. The work presented here is taking the first steps towards modeling techniques that will enable predictive control to execute these actions on command.

Legged locomotion has been modeled using a diverse array of reduced order templates [7]. By applying traditional analyses to these analytic models, predictions of gait characteristics such as speed or stability can be produced. System identification for legged systems has been based on fitting parameters to some of these analytic models. Several papers [1][11][10] fit Spring Loaded Inverted Pendulum (SLIP) model parameters to the stance phase of the RHex family of robots. Bloesch et al. [3] developed a method to fit kinematic parameters and sensor models to the StarlETH, a legged robot. Recently, a parameter fitting approach has been extended to hybrid dynamics by reducing the dimensionality of the system around a periodic orbit [4].

Another technique has been to approximate the rhythmic dynamics of a legged system with a data-driven Floquet model [15]. This work was able to find periodic orbits in a low-dimensional state space to which high frequency disturbances converge. The Floquet models and parameter-fit analytic models can make good predictions for steady-state behavior, but they rely on an assumption of periodic dynamics. They cannot describe transient maneuvers observed when periodic gaits are not enforced. We believe that understanding the full space of behaviors with non-periodic leg motion will be necessary to make headway on useful predictive models for under-actuated yet highly maneuverable legged robots.

To this end, we present a data-driven statistical approach for general modeling of the dynamical behavior of nonlinear systems. Our approach has the benefits of requiring no knowledge of the underlying dynamics, and using computationally simple models to describe the behaviors observed. This identification approach scales well with the dimension of the dynamical state space and number of observations. It is also easy to tune the granularity of the model approximation to make the best use of available computational resources for on-line predictions. The automatic identification of state-space partitioning based on the statistics of the observations can give initial insights into the behavior of a complex dynamical system, which is useful for later analysis.

We assume the dynamics of our millirobots are locally governed by linear dynamics (rigid body motion, kinetic

This material is based upon work supported by the National Science Foundation under IGERT Grant No. DGE-0903711, and Grant No. CNS-0931463, and the United States Army Research Laboratory under the Micro Autonomous Science and Technology Collaborative Technology Alliance.

D.W. Haldane is with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720 USA dhaldane@berkeley.edu

A.D. Buchan, and R.S. Fearing are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA {*abuchan, ronf*}*@eecs.berkeley.edu*

* These authors contributed equally to this work

friction, damped springs) with highly nonlinear shifts in these dynamics (footfalls, four-bar linkage singularities). As such, Piecewise Affine (PWA) systems are a reasonable first approximation of these dynamics. PWA models describe a system as a collection of affine submodels, and a partition of the state-space where each model is applied. These models are very fast to compute for forward prediction, a desirable property for state estimation on low-power systems. Concepts of stability, controllability, and observability have been extended to PWA systems as approximations of nonlinear systems [16], which will enable further analysis with the models generated by this method. In addition, Tedrake et al. [17] have shown that LQR controllers can piece together these regions of state-space for control.

Automatic identification of PWA models is actively being researched [5]. In general, this iterative identification process grows in exponential time with the number of samples, and is thus intractable to compute on very large datasets. Heuristic methods can avoid incurring the computational cost, associated brute-force techniques, but are sensitive to initial region partitioning and can converge to local optimums. As such, we chose simple, statistically-driven clustering methods to identify the partition.

In this paper, we describe current methods for empirically deriving models of dynamical systems. Section II gives the methods for collecting data on our dynamic millirobots and fusing sensor data into a high-fidelity state trajectory. In Section II-E we explain the type of models derived by our method, and the way in which the collected data is partitioned to learn these models. Finally, in Section III we show how the collection of models learned with our method was able to predict future states through simulation of the identified dynamics and control.

II. Methods

This work presents a processing pipeline that identifies several models for a dynamic robotic system, as well as methods for comparing the performance of those models. First, sensor data are collected and fused into a high-fidelity state trajectory (Sections II-A – II-D). Models are then fit to this data, as described in Section II-E. Sec. II-F describes how the models were forward simulated, and Sec. II-G gives our validation method for the model accuracy and simulation. Fig. 4 illustrates this process, and will be referred to in the description of each step of the pipeline.

A. Robotic System

For this analysis, we observed the dynamics of the VelociRoACH [6] running on a treadmill. The VelociRoACH is a hexapedal robot capable of stable running at 27 body-lengths per second with an alternating tripod gait. At the cost of some speed and stability, we did not enforce an alternating tripod gait in order to allow differential drive (diff-drive) steering. In this way, we used the robot as its own disturbance and were able to observe the dynamics of running in a larger region around a stable operating point. It has one actuated degree of freedom per side, a coreless

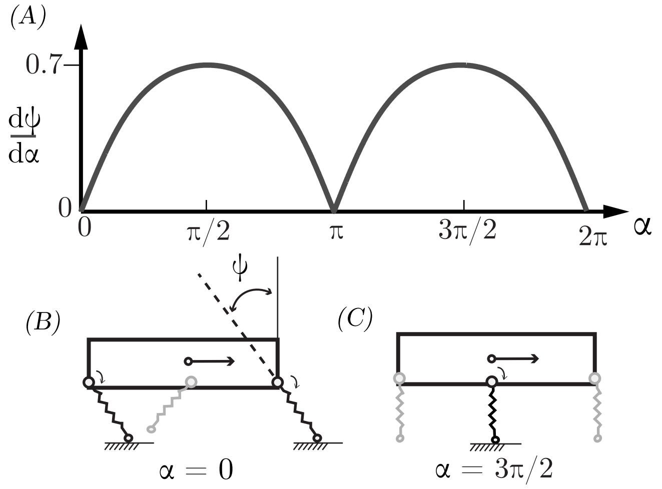


Fig. 2: (A) Change in leg angle, ψ, with respect to motor crank angle, α. Note regions of dead-band around multiples of $\alpha = \pi$. (B) Robot leg configuration at $\alpha = 0$, corresponding to touchdown of the fore and aft legs. (C) Robot leg configuration at $\alpha = \frac{3\pi}{2}$, mid-stance of the middle leg.

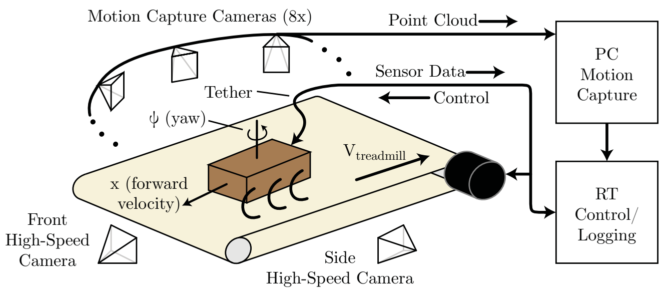


Fig. 3: Experimental setup.

brushed DC motor driving a rotary crank, which drives a set of kinematic linkages. These linkages are made with the Smart Composite Microstructures process [9], and convert rotary motion to leg abduction and adduction. See Haldane et al. [6] for details on mechanical design and dynamic performance. A key feature of this robot is that the leg kinematics for each side are predetermined by the geometry of the kinematic linkages, i.e. there is 1 degree of freedom per side. As the motor crank rotates, the robot takes one step with the fore and aft legs, followed by a step with the middle leg, as shown in Fig. 2. We define one stride to be one full rotation of the motor crank.

A lightweight electronics package[1], controls the VelociRoACH's two motors and streams telemetry data from its on-board sensors. These sensors include a six axis inertial measurement unit (IMU) and 14-bit absolute encoders on the output cranks of each side. The time series of robot sensor information is referred to as $\mathbf{Y}_R(t)$.

B. Experimental Setup

To collect data on dynamic running, we designed a treadmill system with motion tracking and closed-loop position control. The treadmill simulates constant velocity forward running on flat ground, while the motion capture (mocap)

[1]Embedded board: https://github.com/biomimetics/imageproc_pcb

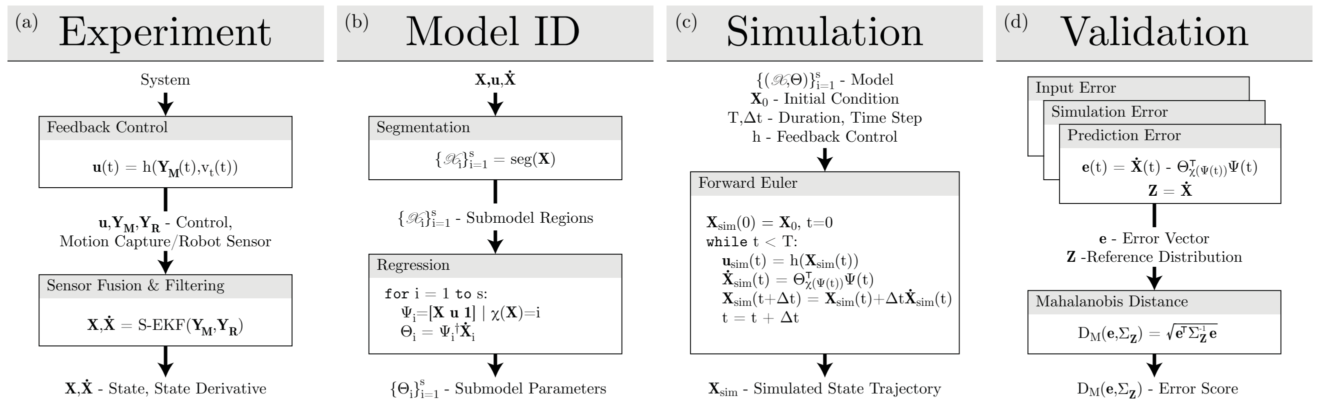


Fig. 4: A block diagram of the method used to learn and test PWA models. Data flows from top to bottom and left to right.

system streams and records robot position information to a real-time control system, which in turn provides motion commands to the robot. High speed video footage was captured from the front and side of the treadmill to observe ground-leg interactions and verify the state trajectory information. Fig. 3 shows the experimental setup.

For this experiment, the VelociRoACH was modified by adding reflective markers and a lightweight tether. The markers enable mocap, and the tether provides power and serial communication to the controller. The combined weight of these modifications is less than 1 g, with negligible effects on the inertia of the robot. The tether did not generate any measurable forces on the robot, and the robot was loaded with a battery for mass. We therefore assume that data collected on the treadmill will be valid for tether-free running on a similar surface.

The treadmill area measures ∼30 cm x 50 cm, allowing 5 robot body lengths of motion in each respective planar axis. A canvas belt provides sufficient traction such that the robot does not slip during nominal running. The belt is driven by a motor with integrated speed controller, allowing the belt to be commanded to speeds up to 5 m/s. Using the motion capture system, we verified that the belt velocity was a linear function of the commanded velocity with a maximum standard deviation of $\sigma = 0.023$ m/s.

The mocap system calculates the position and orientation of the robot $\mathbf{Y}_M = [x\ y\ z\ \psi\ \theta\ \phi]^\top$ at 100 Hz with sub-millimeter resolution and average standard deviations of $\sigma_{xyz} = 0.027$mm and $\sigma_{\psi\theta\phi} = 0.0026$rad, respectively. These data are streamed to a real-time control system, which calculates and streams a control signal to the robot. The position control loop is executed at 100 Hz, using the most recent available data from the mocap system to reduce network latency and jitter. The max round-trip latency was observed to be 30 ms, with an average latency of 10 ms.

To time synchronize the robot, mocap, and video data, an infrared LED on the robot was driven by a square wave originating from the real-time controller. This LED was detected as a marker by the mocap system. The synchronization signal, along with all robot sensor data, are streamed to the real-time controller for logging at 1 kHz. The VelociRoACH has been shown to have significant oscillatory energy in frequencies up to 100 Hz during high speed running [6], which necessitates this high sampling rate.

C. Feedback Control

To control the heading and velocity of the robot, a simple planar differential-drive model was assumed, which had been previously used for another legged millirobot [13]. This clearly ignores much of the complexity of legged locomotion dynamics, but we found it sufficient for keeping the robot on the treadmill long enough to gather data. Our trials were approximately two minutes in length.

Under this model we apply the proportional control in Eqn. (1) to find a desired robot velocity command. Fig. 4a refers to this function as $\mathbf{u} = [\tilde{\dot{x}}\ \tilde{\dot{\psi}}]^\top = h(\mathbf{Y}_M, v_t)$ since it operates directly on mocap sensor information and treadmill velocity v_t.

$$\tilde{\dot{x}} = \frac{v_t - k_x x}{\cos(\psi)} \quad , \qquad \tilde{\dot{\psi}} = -k_\psi \psi \tag{1}$$

Assuming positive v_t and a bounded region of the controlled state-space around $\mathbf{x}_c = [x\ y\ \psi]^\top = [0\ 0\ 0]^\top$, positive gains $k_{x,\psi}$ can be found to guarantee stability for the given region, treadmill velocity. This be shown by considering Lyapunov function $V(\mathbf{x}_c) = ||\mathbf{x}_c||_2^2$. In practice, the control gains were manually tuned to keep the robot stable at each treadmill velocity.

Low level PID control on the position and velocity of the legs is executed at 1 kHz on-board the robot. This control translates a desired robot longitudinal and yaw angular velocity $[\tilde{\dot{x}}\ \tilde{\dot{\psi}}]^\top$ to nominal left and right leg crank angular velocities $[\tilde{\dot{\alpha}}_l\ \tilde{\dot{\alpha}}_r]^\top$. Eqn. (2) shows this conversion for a robot with width d and effective leg radius r.

$$\begin{bmatrix} \tilde{\dot{\alpha}}_l \\ \tilde{\dot{\alpha}}_r \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 1 & d/2 \\ 1 & -d/2 \end{bmatrix} \begin{bmatrix} \tilde{\dot{x}} \\ \tilde{\dot{\psi}} \end{bmatrix} \tag{2}$$

D. Data Fusion and Filtering

In order to study leg function and body motion in dynamic running and turning, we implemented an off-line sensor fusion framework to estimate full robot state. Measurements of position and orientation were collected at 100 Hz using an OptiTrack$^{\text{TM}}$ mocap system. Sensor noise models for the mocap and IMU data were empirically derived by analyzing 1,000 seconds of telemetry data, which were streamed while the robot was stationary. Independent Gaussian noise was assumed on each measurement channel, and the variance for each was subsequently calculated for the sensor measurement model.

The incoming data was filtered for outliers. All observed outliers occurred as mocap tracking errors wherein the state measurement of the robot was miscalculated by several orders of magnitude. These outliers were filtered by rejecting any data point which was more than 4 standard deviations away from the mean. On average, this filter caught 3-4 outliers per 2 minute run. The initial and final twenty strides were clipped from the data set to rule out any transient effect from treadmill start up and slow down. A virtual bound was placed on the treadmill running surface to define a conservative operational area for the robot. If at any time the robot exited this area, a fault condition was set, and all consequent data was discarded from the analysis.

There was a small amount of variance in the sampling period for both the mocap and robot data. A bicubic spline interpolation was used to time shift this irregularly sampled data to the nearest 1 ms time step. The maximum interpolation distance is bounded by one half of the sampling period.

To fuse the data, we used an Extended Kalman Filter (EKF) [18] combined with a minimum-variance smoother (S-EKF). The difference in sampling rates between the robot and the mocap was accounted for by using a time varying observability matrix which limited observations of the mocap state to valid timestamps. We used the motion model developed by Merwe et al. [12] to forward propagate the state. The dynamics of the motion model were linearized and used in the update equations for the EKF. To reduce the variance of the filtered data we performed a backwards pass using a Rauch-Tung-Striebel smoother [14]. The accuracy of the smoothed data was confirmed by applying a known motion profile to a dummy robot, and verified with high-speed video.

The end result of this off-line data processing is a set of high-fidelity state and state derivative trajectories $\mathbf{X}$ and $\dot{\mathbf{X}}$. The 16-dimensional state vector $\mathbf{X}$ consists of the first-order position variables $\mathbf{q}$ and their continuous-time derivatives $\dot{\mathbf{q}}$. Shown in Eqn. 3, $\mathbf{q}$ contains the position and orientation of the robot body, and the crank angles (α_r, α_l) of each leg mechanism. Robot position and orientation are recorded in the world frame, while all other variables are considered in robot-fixed axes. The highest order physical variables are accelerations in the $\ddot{\mathbf{q}}$ portion of $\dot{\mathbf{X}}$.

$$\mathbf{q} = \begin{bmatrix} x & y & z & \psi & \theta & \phi & \alpha_l & \alpha_r \end{bmatrix}^\top, \tag{3}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}, \dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} \tag{4}$$

E. Model Identification

The models identified in this work are time-invariant piecewise affine state-space differential equations with exogenous inputs. Functionally this means a model maps a state and input to a single estimated continuous-time state derivative. For a fixed submodel, this map in classical linear system terms is: a system matrix A, input matrix B, with an affine forcing vector f. For convenience we concatenate A,B, and f to a parameter matrix Θ_i, and the state, input, and affine component to a regressor vector $\Psi(t)$. Equation 5 describes this map to state derivative prediction, $\hat{\dot{\mathbf{X}}}(t)$.

$$\hat{\dot{\mathbf{X}}}(t) = \begin{bmatrix} A \; B \; f \end{bmatrix}_i \begin{bmatrix} \mathbf{X}(t) \\ \mathbf{u}(t) \\ 1 \end{bmatrix} = \Theta_i^\top \Psi(t) \tag{5}$$

Submodel parameters are indexed by i, which indicates the submodel region $\mathscr{X}_i \subset \mathbb{R}^n$ in which those dynamics hold. The regions are disjoint $(\forall i \neq j, \mathscr{X}_i \cap \mathscr{X}_j = \varnothing)$, and complete $(\bigcup_i \mathscr{X}_i = \mathbb{R}^n)$. A complete model consists of the collection of tuples of submodel parameters and regions $\{(\Theta, \mathscr{X})_i\}_{i=1}^s$, where s is the number of submodels. As Fig. 4b illustrates, our model identification technique first partitions the space via statistical segmentation methods, then finds the submodel parameters using linear regression.

1) Segmentation: In a practical sense, we are interested in a balance between the number of model partitions and the overall accuracy of the estimation. To explore this relationship, we evaluated three different methods of partitioning: "Average", "z-score", and "k-means". To test the null hypothesis of no linear dynamics, we also define a null model $[A \; B \; f]_{\text{null}} = [0 \; 0 \; \mu]$, where μ is the average value of $\dot{\mathbf{X}}_i$. We use this collection of automatic region identification methods tests values of $s = [0, 1, 2, 10, 50]$.

The Average method considers all observed data in one region, and thus identifies one affine model for the entire system. This approach is a base case for a dynamical system.

The z-score method divides the observations into two partitions based on the empirical likelihood of the observation. First the data are z-score normalized, giving the distribution of observations mean 0 and variance 1 in each axis. A parameter σ_z is used to separate the data. Observations within σ_z of the origin in z-score space are considered one region, and the remaining data outside this ball are a second region. This approach is primarily used to see if a submodel fit to the most likely observations improves the overall model fit. We tested the z-score model with $\sigma_z = 1.5$ (listed as Z-1.5), which empirically improved the fit of the inner region over the outer region by an average of 25% per axis.

To extend this method to more partitions, we used the k-means clustering method. First, all data in $\mathbf{X}$ are z-score normalized. Then, standard k-means is applied to the normalized data to identify $k = s$ model partitions. We evaluated this approach for s = 10 and s = 50 to define the K-10 and K-50 models. Varying s allows tuning of the model granularity; we chose a maximum of s=50 so that the partition could be computed in a few minutes. Empirically we found that the number of observations in each region was approximately uniform for each trial.

For convenience, we define the submodel selecting function χ in Eqn. 6, which maps a state vector to the submodel region index i that the state portion of $\mathbf{X}$ resides in. We also use the notation $\chi(\Psi)$, which is understood to act only on the $\mathbf{X}$ portion of Ψ. Where relevant, the subscript M denotes which model partition is used.

$$\chi : \mathbb{R}^n \to \{1, ..., s\}$$

$$\chi_M(\mathbf{X}(t)) = i \quad \text{s.t.} \quad \mathbf{X}(t) \in \mathscr{X}_i \tag{6}$$

In practice the submodel regions can be defined explicitly with polytopic separating planes, or implicitly by other methods. In the case of z-score and k-means segmentation algorithms, χ can be more simply evaluated by storing the parameters of the model and calculating the regions of a new observation from those. For example, by storing the means of the k-means segmentation method as $\{\mu_i\}_{i=1}^{s}$, the region index can be calculated as:

$$\chi_{K-s}(\mathbf{X}) = \underset{i \in \{1,...,s\}}{\arg\min} \; ||\mathbf{X} - \mu_i||_2 \tag{7}$$

We use this approach due to simplify the representation in the simulation results discussed below. In general any method that reproduces the complete and disjoint mapping will suffice.

2) Regression: Once the data are segmented, we estimate the dynamics of each region of the observed data using least squares. Given a collection of correlated observations, the learned models are an estimator of $\dot{\mathbf{X}}(t)$, given $\mathbf{X}(t)$ and $\mathbf{u}(t)$. The estimation error $\mathbf{e}(t)$ for a particular observed $\dot{\mathbf{X}}(t)$ and model is therefore:

$$\mathbf{e}(t) = \dot{\mathbf{X}}(t) - \Theta^{\top}_{\chi(\Psi(t))} \Psi(t) \tag{8}$$

By collecting all observations in a region $\mathscr{X}_i$ as column vectors in Ψ_i, we can calculate the submodel parameters that minimize the region sum squared prediction error in $\mathbf{e}_i$ as Eqn. 9, where the dagger represents the Moore-Penrose pseudoinverse.

$$\Theta_i = \Psi_i^{\dagger} \dot{\mathbf{X}}_i \tag{9}$$

F. Simulation

The PWA models identified by this approach can be used to simulate a system state trajectory (Fig. 4c). The simulations presented in this work use naïve Euler integration with exogenous control and state determined submodels (Eqn. 7) to generate trajectories. At each time step, the control $\mathbf{u}_{sim}$ is calculated from $\mathbf{X}_{sim}$ using the same control law discussed in II-C. The submodel parameters Θ_i used to calculate the state derivative are chosen such that $i = \chi(\mathbf{X}_{sim})$.

TABLE I: Model Comparison

	Variance Normalized Average Error							
Model	$\ddot{x}$	$\ddot{y}$	$\ddot{z}$	$\ddot{\psi}$	$\ddot{\theta}$	$\ddot{\phi}$	$\ddot{\alpha_L}$	$\ddot{\alpha_R}$
Null	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Avg	0.843	0.781	0.672	0.950	0.790	0.737	0.820	0.821
Z-1.5	0.791	0.760	0.622	0.925	0.768	0.730	0.744	0.765
K-10	0.719	0.728	0.594	0.839	0.648	0.663	0.727	0.713
K-50	0.627	0.638	0.521	0.684	0.516	0.529	0.575	0.555
$\boldsymbol{\sigma}$	6.581	10.18	9.449	515.9	294.7	112.1	2259	2203
$units$	m/s^2	m/s^2	m/s^2	rad/s^2	rad/s^2	rad/s^2	rad/s^2	rad/s^2

G. Validation

Comparing the performance of these models requires a metric that relates vector quantities in different spaces based on the distribution of values in that signal. We chose the Mahalanobis Distance (D_M) of an error vector $\mathbf{e}$ with respect to collection of observations $\mathbf{Z}$. $\Sigma_{\mathbf{Z}}$ is the covariance matrix of the distribution of $\mathbf{Z}$.

$$D_M(\mathbf{e}, \Sigma_{\mathbf{Z}}) = \sqrt{\mathbf{e}^{\top} \Sigma_{\mathbf{Z}}^{-1} \mathbf{e}} \tag{10}$$

The remainder of our discussion will refer to a vector $\mathbf{e}$ as an "error", and the scalar value calculated by D_M as a "score". A higher score corresponds to a greater error, and thus a poorer prediction.

III. Results and Discussion

Our results are presented in three major sections. The first describes how the models perform strictly as a predictor of the state acceleration based on the state and input. Next we evaluate how forward simulation of state trajectories varies across the modeling approaches and duration of simulation. Finally we examine how the state-space partitions identified by segmentation may be correlated with physical nonlinearities.

A. Data Selection

Our approach to model identification relies on collecting data from a robot running in a stable or desirable operating regime. Models fit to this zone of operation would then be most accurate around a nominal point of stable running. We collected running data from the VelociRoACH with treadmill speeds in the range of 0.1 to 0.5 m/s. Fig. 5 shows the resulting distribution of leg phase observations as a function of treadmill speed. Leg phase is defined to be $\Phi = \alpha_l - \alpha_r + \pi$. The most stable operating speed was 0.25 m/s, corresponding to a stride frequency of approximately 5 Hz. At this leg frequency, the robot tends to passively converge to an alternating tripod gait ($\Phi = 0$) which indicates an empirical basin of stability for the alternating tripod gait in this region.

B. Model Evaluation

All of the models discussed in this work are predictors of state derivative, conditioned on state and control input. We have constrained the parameters of our identification to second-order state-space differential equations and so our

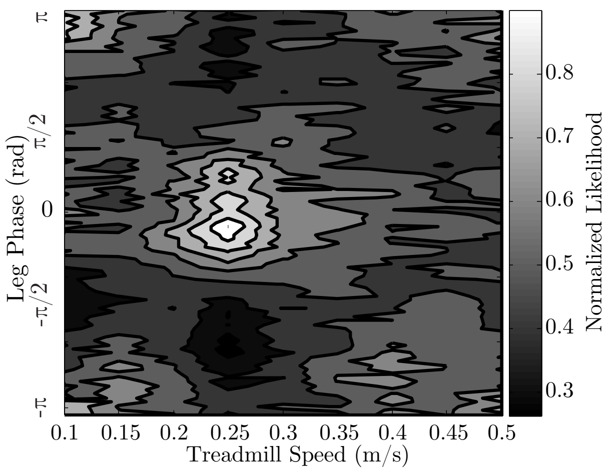


Fig. 5: Contour map of probability of leg phase plotted against velocity. Lighter regions are more likely to occur. There are approximately 100,000 observations for each measured speed.

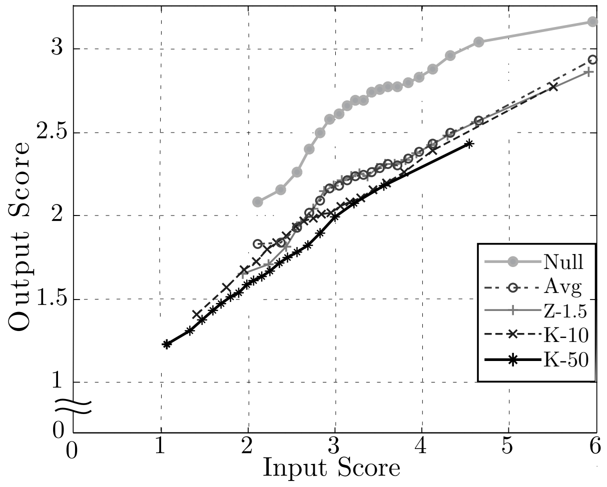


Fig. 6: Model performance on an Input/Output basis, as a function of approach. Input score is defined as $D_M(C_{\chi(\mathbf{X}(t))} - \mathbf{X}(t), \Sigma_\mathbf{X})$ where $C_{\chi(\mathbf{X}(t))}$ is the centroid of the submodel region containing $\mathbf{X}(t)$. Output score is defined as $D_M(\dot{\mathbf{X}} - \hat{\dot{\mathbf{X}}}, \Sigma_{\dot{\mathbf{X}}})$. The marked points on each series are the mean of an equal number of observations, so that the density of the points indicates the distribution of observations on each axis.

prediction output space is the acceleration vector, $\ddot{\mathbf{q}}$. Table I reports the average prediction error for each of these variables for all five of the modeling approaches we consider. These results were generated using 10-fold cross-validation between model generation and prediction. The table values are the average absolute error between the predicted and actual value, normalized by the standard deviation of that value. The standard deviation of each variable is reported in the last row to provide scale.

The Null model shows a baseline normalized error of 1 for all variables. The prediction accuracy improves as the number of model regions increases. This validates the hypothesis that a linear model at least improves over the Null model with no dynamics, and that more submodel regions allow better local approximations of non-linear dynamics.

Fig. 6 shows that as more regions are added, the average distance to a submodel centroid decreases. This empirically

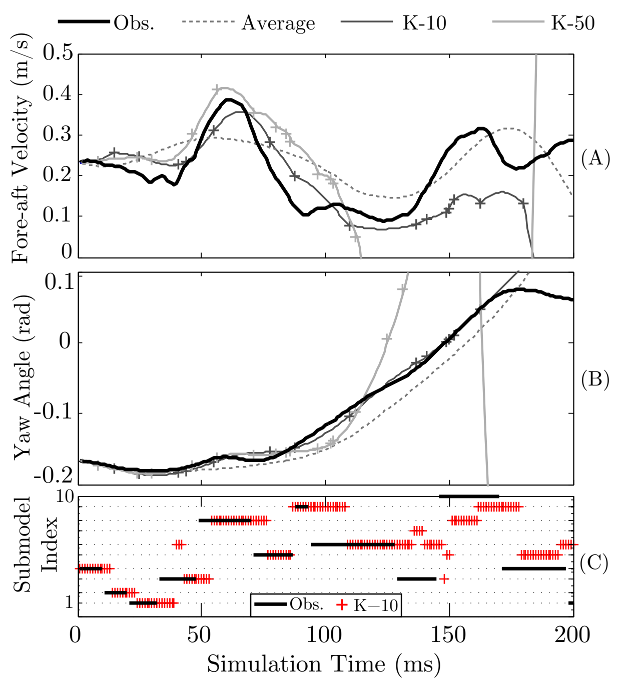


Fig. 7: Predicted time trajectories of fore-aft velocity (A), Yaw angle (B), and the submodel index of the K-10 simulation state $\chi_{K-10}(X_{K-10})$ superimposed on the K-10 submodel index of the observed trajectory $\chi_{K-10}(X_{obs})$ (C). In (A) and (B), transitions between submodels are marked with + symbols.

shows the intuitive positive correlation between input score and output score. We expect that an input observation close to the centroid of a subregion is representative of the behavior in that region, and thus the output prediction is best in that vicinity.

C. Simulation

As discussed in section II-F, the models can be used to predict the behavior of the system. Fig. 7 shows characteristic simulations for several models compared to the experimentally observed trajectory of the system. We chose to show forward velocity ($\dot{x}$) and yaw (ϕ), which are examples of first and zeroth-order state predictions. These variables are also useful to predict robot motion for turning maneuver planning. In each case we initialized the simulation state to an observed state, and then simulated the state and control for 200 1 ms time steps. The observed data for the duration of these simulation times was excluded from the data used to train the models. Fig. 7 (A) and (B) show simulated trajectories of the Average, K-10, and K-50 models, along with the observed trajectory of the system.

The simulations qualitatively follow the behavior of the system through several submodel regions. After a simulation time horizon of 100-125 ms the model prediction of the state diverges from the observed state trajectory. Fig. 7 (C) shows that the submodel transitions in the simulation lag the observed submodel transitions. This lag causes a divergence from the observed submodel region trajectory for the K-10 model. The dynamics of the K-50 model are noisier than those of K-10 due to a larger number of transitions between

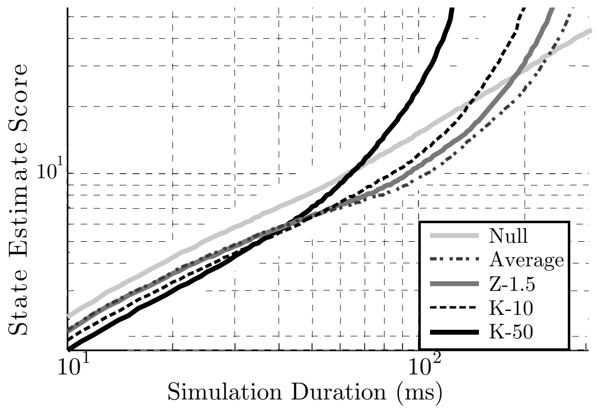


Fig. 8: This plot was generated using 1000 simulations with initial conditions randomly selected from $\mathbf{X}$. State Estimate Score is defined as $D_M(\mathbf{X}(t_0 + t_{sim}) - \mathbf{X}_{sim}(t_{sim}), \Sigma_{\mathbf{X}})$.

the affine submodels. The K-50 submodel trajectory diverges earlier than K-10 which shortens the time horizon of its usefulness.

The time horizon of useful predictions from simulations decays with the number for submodels. Figure 8 shows this trend of prediction accuracy versus simulation duration. The plot shows a Pareto frontier for the model complexity. The K-50 model is most accurate for short time spans (up to 40 ms), K-10 for a brief midrange (40-60 ms), and the average model remains closest to the observed trajectory from 60 to 200 ms. After 200 ms, all models make a poorer estimate than a null model with constant acceleration.

D. Interpretation of Model Regions

A hypothesis of our model identification approach is that model regions will be roughly associated with distinct dynamic behaviors which are caused by physical nonlinearities. To explore how the models identified for our robot correspond to a physically interpretable dynamic structure, we project the likelihood of model transition onto a plane of the state-space where we expect regions of disparate dynamics to be readily identifiable.

The leg crank angle largely determines whether a leg is on or off the ground, so we expect it to generate the clearest distinctions in dynamics. To examine this hypothesis, we analyze how closely the observed dynamics match a simple diff-drive kinematic model as a function of the projection to the leg phase space (the plane of (α_l, α_r)). We consider the diff-drive state $\mathbf{X}_D = [\dot{x} \ \dot{\psi}]^\top$. The kinematic model predicts that $\mathbf{X}_D$ is the inverse of the linear mapping in Eqn. 2; we label this prediction $\hat{\mathbf{X}}_D$. In Fig. 9a we see that the diff-drive model does not make accurate predictions of robot behavior over the course of a stride. Variations in the local dynamics of the robot from the kinematic model are shown by the color in this figure. Regions of high contrast indicate that the dynamics of a region are changing rapidly. These changes were caused by leg touchdown and liftoff events, physical nonlinearities which ideally would be matched by submodel transitions.

Fig. 9b shows submodel transition ratio projected onto the same leg angle space. The transition ratio is calculated by binning the state trajectories on this space, and reporting the ratio of trajectories that experience a submodel transition to the total number of trajectories in the bin.

If there were no structure to the observed state trajectories, then we would expect there to be an even distribution of submodel transitions. From a mechanical perspective, we would expect leg touchdown events to be hybrid transitions in the robot dynamics. The most salient trend visible in Fig. 9b are the high probability bands in the vicinity of $3\pi/4$, which corresponds to liftoff of the front and rear legs (see Fig. 2). These bands match well with several regions of high-contrast in Fig. 9a.

IV. Conclusions and Future Work

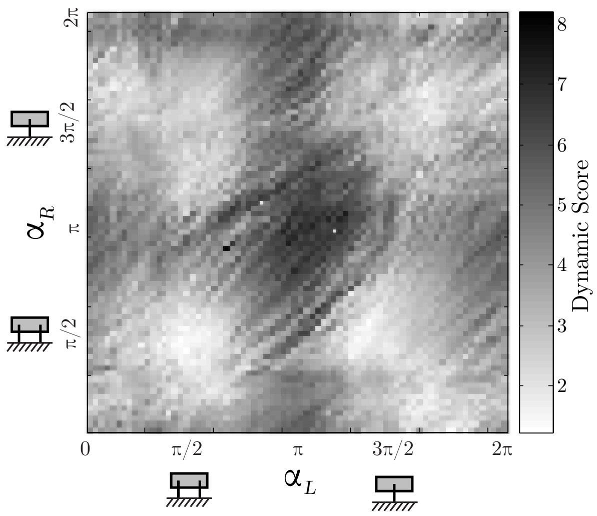
We applied a highly simplified differential drive model to a legged robot, which fit well enough to control its position and heading on a treadmill up to speeds of 5 body-lengths per second. This allowed for the collection of a large dataset of robot state observations. We demonstrated a data-driven methodology which learns models that predict the observed robot 16 dimensional state and derivative on time scales of about one stride. The methodology is platform agnostic, which will allow application of our approach to determine the dynamics of a wide variety of systems.

The predictive ability of the sets was measured via a Mahalanobis Distance metric (D_M). The one-time-step predictive performance of learned models increased with the granularity of the state space partitions. However, forward simulations of models with more subregions tended to diverge from the observed state trajectory in shorter time periods than simulations of simpler models. We expect this balance between short and long-term predictions to be a trend in all models identified with this technique. The choice of model granularity will depend on the prediction and simulation goals of a specific robotic state estimation problem.

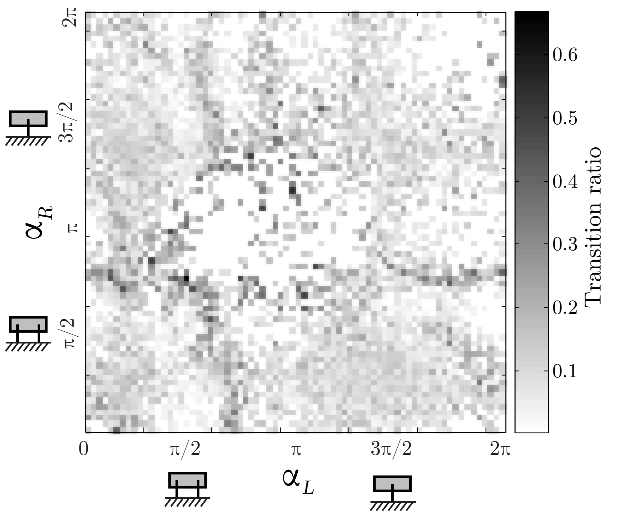
Future work will target further improving the predictive ability of the data derived models. More sophisticated partitioning strategies could be employed by allowing the shape of the k-means clusters to more closely match the dynamical transitions of the robot. Iterative optimization techniques could then modify the regions to move towards improving the descriptive ability of the models with fewer model subregions. Analysis under the model selection framework of the Bayesian Information Criterion (BIC) could help automatically tune the balance of model complexity and prediction.

Simulation could also be improved over naïve Euler integration. Probabilistic frameworks such as Gaussian mixture models and Markov Chains could add information about the likelihood of model transitions, and reduce the instability seen due to diverging from observed submodel index trajectories.

Finally, we are planning future work on using these learned models for on-line state estimation and control. The models are particularly amenable to an on-board Kalman filter or particle filter, as the simple to compute affine



(a) Heat-map of differential drive dynamics observed for the robot. The color axis shows the Dynamic Score, defined as $D_M(\mathbf{X}_D - \hat{\mathbf{X}}_D, \Sigma_{\mathbf{X}_D})$,projected onto leg phase space.



(b) Heat-map of model transition ratio as a function of left and right crank angle (α_L, α_R) for the K-50 model.

Fig. 9: Heat-maps showing observed robot and model dynamics. The robot stick figures on the axes show how mid-stance for each of the legs is associated with the α variable. Top dead center for the front and rear legs occurs at $\pi/2$, and at $3\pi/2$ for the middle leg

dynamical models can also be stored with an estimate of their accuracy. This relationship can be easily approximated with the Input/Output score relationship in Fig. 6, or extended to a full empirical motion model covariance. Extending to control and planning, other future investigations could use the models to investigate the dynamics of aggressive maneuvers, such as rapid turns.

With these additions, the work presented here would be a first step to a modeling and control paradigm that could be used on nearly any type of dynamical system. If state trajectories in regions of interest can be explored, our approach can identify a collection of affine models that predict the dynamics with tunable granularity. These models can then identify physical parameters of interest, provide a variable horizon simulation of the system, or provide empirical measures of prediction uncertain in probabilistic planning frameworks.

Acknowledgments

The authors would like to thank Ben Morse and Dave Rollinson for their help with Kalman filtering. Thanks also to the Biomimetic Millisystems Lab for their helpful comments. Thanks to Sam Burden, Henrik Ohlsson, and Roy Dong for discussions on Hybrid Systems.

References

[1] R. Altendorfer, D. E. Koditschek, and P. J. Holmes, “Stability Analysis of a Clock-Driven Rigid-Body SLIP Model for RHex,” *The International Journal of Robotics Research*, vol. 23, no. 10-11, pp. 1001–1012, Oct. 2004.

[2] P. Birkmeyer, K. Peterson, and R. S. Fearing, “DASH : A dynamic 16g hexapedal robot,” *IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 2683–2689, 2009.

[3] M. Bloesch, M. Hutter, M. A. Hoepflinger, C. Gehring, and R. Siegwart, “Kinematic Batch Calibration for Legged Robots,” *IEEE Int. Conf. on Robotics and Automation*, no. 3, pp. 2527–2532.

[4] S. Burden, S. Revzen, and S. Sastry, “Dimension reduction near periodic orbits of hybrid systems,” *IEEE Conf. on Decision and Control*, pp. 6116–6121, 2011.

[5] G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari, “A clustering technique for the identification of piecewise affine systems,” *Automatica*, vol. 39, no. 2, pp. 205–217, Feb. 2003.

[6] D. W. Haldane, K. C. Peterson, F. L. Garcia Bermudez, and R. S. Fearing, “Animal-inspired Design and Aerodynamic Stabilization of a Hexapedal Millirobot,” *IEEE Int. Conf. on Robotics and Automation*, pp. 3264–3271.

[7] P. Holmes, R. J. Full, D. E. Koditschek, and J. Guckenheimer, “The Dynamics of Legged Locomotion: Models, Analyses, and Challenges,” *SIAM Review*, vol. 48, no. 2, pp. 207–304, Jan. 2006.

[8] A. M. Hoover, S. Burden, X. Y. Fu, S. S. Sastry, and R. S. Fearing, “Bio-inspired design and dynamic maneuverability of a minimally actuated six-legged robot,” in *3rd IEEE RAS and EMBS Int. Conf. on Biomedical Robotics and Biomechatronics (BioRob)*, Sep. 2010, pp. 869–876.

[9] A. M. Hoover and R. S. Fearing, “Fast scale prototyping for folded millirobots,” *IEEE Int. Conf. on Robotics and Automation*, pp. 886–892, 2008.

[10] H. Komsuoglu, A. Majumdar, Y. O. Aydin, and D. E. Koditschek, “Characterization of Dynamic Behaviors in a Hexapod Robot,” *Int. Symposium on Experimental Robotics*, 2010.

[11] H. Komsuoglu, K. Sohn, R. J. Full, and D. E. Koditschek, “A physical model for dynamical arthropod running on level ground,” *Int. Symposium on Experimental Robotics*, pp. 303–317, 2008.

[12] R. V. D. Merwe, E. A. Wan, and S. I. Julier, “Nonlinear Estimation and Sensor-Fusion - Applications to Integrated Navigation -,” *Proc. AIAA Guidance Navigation and Controls Conf*, pp. 1–30, 2004.

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

[14] H. E. Rauch, F. Tung, and C. T. Striebel, “Maximum likelihood estimates of linear dynamic systems,” *J. Amer. Inst. Aeronautics and Astronautics*, vol. 3, no. 8, pp. 1445–1450, Aug. 1965.

[15] S. Revzen and J. M. Guckenheimer, “Finding the dimension of slow dynamics in a rhythmic system.” *Journal of the Royal Society, Interface / the Royal Society*, vol. 9, no. 70, pp. 957–71, May 2012.

[16] E. Sontag, “Nonlinear regulation: The piecewise linear approach,” *IEEE Transactions on Automatic Control*, pp. 346 – 358, 1981.

[17] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, “LQR-trees: Feedback Motion Planning via Sums-of-Squares Verification,” *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, Apr. 2010.

[18] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, 2005.