Learning and planning high-dimensional physical trajectories via structured Lagrangians

Paul Vernaza, Daniel D. Lee, Seung-Joon Yi
GRASP Laboratory
University of Pennsylvania, Philadelphia, PA 19104
{vernaza,ddlee,yiseung}@seas.upenn.edu

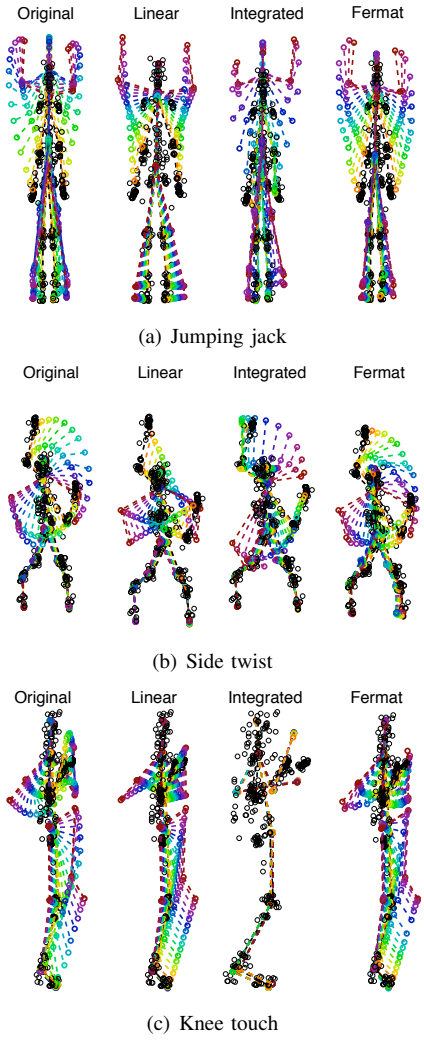
***Abstract*— We consider the problem of finding sufficiently simple models of high-dimensional physical systems that are consistent with observed trajectories, and using these models to synthesize new trajectories. Our approach models physical trajectories as least-time trajectories realized by free particles moving along the geodesics of a curved manifold, reminiscent of the way light rays obey Fermat's principle of least time. Finding these trajectories, unfortunately, requires finding a minimum-cost path in a high-dimensional space, which is generally a computationally intractable problem. In this work we show that this high-dimensional planning problem can often be solved nearly optimally in practice via deterministic search, as long as we can find a certain low-dimensional structure in the Lagrangian that describes our observed trajectories. This low-dimensional structure additionally makes it feasible to learn an estimate of a Lagrangian that is consistent with the observed trajectories, thus allowing us to present a complete approach for learning from and predicting high-dimensional physical motion sequences. We finally show experimental results applying our method to human motion and robotic walking gaits. In doing so, we furthermore demonstrate efficient path planning in a 990-dimensional space.**

I. Introduction

In general terms, our interest in this work is to learn as much as possible about a physical system from observed trajectories of the system. We would then like to use what we have learned to generate new plausible trajectories that obey certain boundary conditions; i.e., each trajectory should start at a specified point, and end at another specified point.

In practical terms, Figure 1 illustrates the kind of problem we might want to solve. In this example, a motion capture system tracks the positions of 330 reflective markers placed at specific locations on the body of a human performing various exercises. Given sequences of this form as training data, we now wish to specify two new poses, and deduce a sequence that plausibly interpolates between the two. For example, if the subject is standing with his hands at his sides in the first pose, and he is standing with his hands over his head in the second pose, the training data might suggest that we interpolate a sequence resembling a jumping jack between them.

Our approach models the observed trajectories as being least-time trajectories of what we describe as a *kinetic* Lagrangian system. We define a kinetic Lagrangian system to be one in which the observed D-dimensional trajectories



(a) Jumping jack

(b) Side twist

(c) Knee touch

Fig. 1. Visualization of experiments in producing novel trajectories from pairs of key frames. Leftmost image in each set is the true, held-out trajectory. Other images depict the output of different learning methods applied to reconstruct the trajectory from the key frames, as described in Section III.

$x(t) \in \mathbb{R}^D$ are extremal with respect to the cost functional

$$J\{x\} = \int_{t_1}^{t_2} \frac{1}{2} m(x) \|\dot{x}\|^2 dt \tag{1}$$

over all differentiable trajectories that intersect $x(t_1)$ and $x(t_2)$. Learning the parameters of this model is equivalent to learning the function $m(x)$, which has physical interpretations that will be described later in this work. Once we have learned $m(x)$, a feasible trajectory interpolating two new states can be found by minimizing (1) over all trajectories $x(t)$.

Although this is an accurate description of our method, we have not yet mentioned the critical issue of the curse of dimensionality. In practice, minimizing (1) over all trajectories $x(t)$ takes time exponential in D. For a path planning problem with $D > 3$ or 4, one typically first resorts to deterministic heuristic search, such as A* [1] or weighted A*. For larger D, randomized planning algorithms, such as Probabilistic Roadmaps (PRM) [2] and Rapidly-exploring Random Trees (RRT) [3] are often useful, though they are better suited for feasibility problems than cost minimization. More recently, R*, a hybrid randomized-deterministic planning algorithm has been demonstrated to effectively minimize cost in a practical problem with $D \sim 10$ [4]. Unfortunately, in our application with human motion capture data, $D = 990$. Clearly, no generic path planning algorithm would be suitable for this task.

Moreover, learning the parameters of (1) (i.e., the map $m(x) : \mathbb{R}^{990} \to \mathbb{R}$) is an equally-impossible task if attempted directly, due to the well-known curse of dimensionality in machine learning [5].

In this work, we give methods to effectively solve the high-dimensional path planning and learning problems as long as we make the critical assumption that the cost function for our path planner (i.e., (1)) possesses a hidden low-dimensional structure. If it does possess this structure, or it possesses this structure approximately, we give robust algorithms to discover it, leverage it to solve the learning problem, and finally solve the high-dimensional planning problem in a nearly-optimal way, given certain technical assumptions.

A. Related work

Some similarities might be drawn between our method and system identification methods in the controls literature [6]. However, our approach distinguishes itself from most system identification methods by introducing low-dimensional structure to make the learning and inference problems tractable for high-dimensional systems. In this way, our method is similar to LWPR [7], a regression algorithm that learns representations that are locally low-dimensional in order to deal with high-dimensional systems. However, LWPR does not assume a physical basis for the observed data. We show how making this physical assumption leads to a natural class of low-dimensional models that describe physical systems particularly well, in addition to admitting practical learning and inference algorithms.

Our method also shares some similarities with the recent work of Sekimoto et. al. [8], which demonstrates that some human motions are well-modeled by a purely kinetic Lagrangian model with no external forces applied. However, their method assumes that an accurate physical model of humanoid motion is already given. Our method learns a model using only observed trajectories, and can be applied to any physical system just as easily as it is applied to humanoid motion. We also note that [8] makes no provision for the problems that arise with very high-dimensional problems.

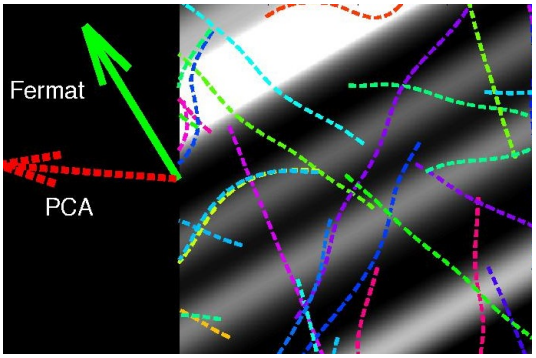
We were partially inspired by recent developments in the learning community that have focused on generating low-dimensional models of dynamical systems. In particular, Gaussian Process Dynamical Models [9] (GPDM) model high-dimensional observed trajectories as images of low-dimensional latent trajectories under an unknown stochastic mapping, while another unknown stochastic mapping serves as a Markov transition function between latent states. These maps are marginalized over in a Bayesian setting to infer a probabilistic joint distribution between latent and observed trajectories.

The crucial difference between our method and assorted Gaussian process latent variable models [10] is that the latter fundamentally assume that the observed state itself has an underlying low-dimensional structure induced by the fixed mapping from latent to observation spaces, while our method assumes low-dimensional dependence structure of the Lagrangian. We would argue that the latter assumption is a more meaningful assumption to make when one knows that the system under consideration is physical.

Our work also bears some similarity to maximum margin planning [11] (MMP), which might be considered to be the inverse problem of finding shortest paths on a weighted graph (or Markov decision process), with the further restriction that the graph weights must be linear combinations of given features. Although our method also incorporates an inverse shortest path problem, we consider the case of a continuous state space as opposed to the discrete state space of MMP. This allows us to reason about a structured cost function in a straightforward way—a structured cost function ($m(x)$, in our setting) is one that depends on a small number of state variables. By contrast, MMP induces no low-dimensional structure on the state space, partially because it is not as straightforward to speak of the low-dimensional structure of the combinatorial object (i.e, graph) that is the state space involved in MMP. A practical consequence of this distinction is that MMP cannot easily be applied to Markov decision processes with high-dimensional state spaces, due to the intractability of the associated planning problem.

II. The method of Fermat components

The description of our method proceeds in several parts. We begin with a brief description of our model and its physical meaning. We then derive a simple algorithm, which we call Fermat Component Analysis (FCA), that identifies the low-dimensional structure of the Lagrangian from observed trajectory data. Next, we describe our approach to estimating



(a) Simulated light rays and discovered directions

(b) Learned cost function: FCA

(c) Learned cost function: PCA

Fig. 2. Simple demonstration of Fermat components applied to a kinetic Lagrangian system: light rays (dashed lines in 2(a)) traveling through a medium of varying refractive index. Brigher areas have higher refractive index (equivalently, higher cost, or lower speed). Light rays turn towards high-cost regions upon entering them to minimize travel time, in accordance with Fermat's principle of least time. Solid arrow shows the first Fermat component of the data. Dashed arrow shows the first PCA component of the data. 2(b) shows the underlying one-dimensional cost function learned by performing regression of the cost function from the Fermat component, and 2(c) shows the result of regression from the PCA component.

the associated structured Lagrangian. With an estimate of the Lagrangian, we finally describe how we perform path planning in the original high-dimensional space in order to produce new trajectories.

A. Kinetic Lagrangian systems

As mentioned in the introduction, our model consists of the assumption that each observed trajectory $x(t)$ is extremal with respect to the cost functional (1). We now make some remarks about (1) in order to elucidate the nature of our model.

First, we observe that the kinetic energy T of a physical system with (time-invariant) generalized coordinates x can always be written as a quadratic form: [12]

$$T = \frac{1}{2}\dot{x}^T[M(x)]\dot{x} \tag{2}$$

We can identify this form with the integrand in (1) to obtain the relationship

$$M(x) = m(x)I \tag{3}$$

where I is the identity matrix. Therefore, we can think of (1) as the *action* of a system possessing no potential energy (i.e., no external forces), and whose kinetic energy is given by a diagonal quadratic form in generalized coordinates.

Furthermore, note that we can construct a Riemannian manifold with $M(x)$ as its metric tensor. The geodesics of this manifold are the extremal paths $x(t)$ of the cost functional

$$J'\{x\} = \int_{t_1}^{t_2} \sqrt{\dot{x}M\dot{x}}dt = \int_{t_1}^{t_2} \sqrt{m(x)}\|\dot{x}\|dt \tag{4}$$

By a trivial application of well-known result [12], the extremal paths of (4) and (1) are actually the same; i.e., the paths taken by the physical system with action given by (1), are actually geodesics of the Riemannian manifold with metric $M(x)$.

Finally, we assume that the observed trajectories are least-time trajectories, in which case we have

$$m(x) \propto \frac{1}{\|\dot{x}\|^2}. \tag{5}$$

Note that this assumption effectively allows us to observe the values of $m(x)$ along the observed trajectories, which will be useful later on.

A canonical example of our model is therefore the propagation of light rays in optics, which is governed by Fermat's principle of least time (Figure 2). In this model, each point in space is assigned a speed at which light is allowed to propagate through the point; the inverse of this speed (i.e., the index of refraction) is equal to $\sqrt{m(x)}$ in our model.

B. Fermat Component Analysis

We now wish to analyze the behavior of our system when $m(x)$ possesses a low-dimensional structure. Once we know the expected behavior in this case, we can exploit it to detect the low-dimensional structure, if present.

We have already mentioned how physical trajectories have a macroscopic description as those paths that make the action locally extremal. However, they also have a local description in the form of the Euler-Lagrange equations [13] of the calculus of variations. In our case, letting $L = \|\dot{x}\|\sqrt{m(x)}$ (c.f. (4)), these take the form

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = 0 \tag{6}$$

Denote by x_i the ith component of the state vector, and suppose that $m(x)$ does not depend on x_i. For notational convenience, we write

$$C(x) = \sqrt{m(x)} \tag{7}$$

Since $\partial C/\partial x_i = 0$, it follows from the Euler-Lagrange equations that

$$\frac{\dot{x}_i}{\|\dot{x}\|}C(x) = k \tag{8}$$

where k is a constant that depends on the initial conditions of the trajectory. The failure of this *conservation law* to hold for some variable x_i implies that $m(x)$ must depend

on x_i. This is the fundamental behavior we exploit to detect the low-dimensional structure of $m(x)$. Recalling the example of light propagation, these conservation laws might be considered a multidimensional generalization of Snell's law [14].

Assume now that instead of observing $x(t)$ directly, we observe a trajectory $z(t) = Ux(t)$, where U is an a-priori unknown square orthonormal matrix. The previous conservation law can then be rewritten as

$$\left(u^T \frac{\dot{z}}{\|\dot{z}\|}\right) C(z) = k \tag{9}$$

where u is some column of U.

Note that the only unknowns in the conservation law above are u and k. If we wish to find a projection that does not satisfy the conservation law, it seems reasonable to find a projection that maximizes the inequality of this expression. Given our assumption that $C(x)$ is given by inverse speed, and assuming we are given discrete samples $z_{[i]}$ of the trajectory and its derivative, we therefore propose the following optimization to find u:

$$\max_{\|u\|=1} \min_{k_1 \ldots k_T} \sum_i \left(u^T \dot{z}_{[i]} - k_{p_i} \|\dot{z}_{[i]}\|^2\right)^2 \tag{10}$$

Note that we have introduced additional k variables in order to account for the possibility that each sample may be taken from one of T trajectories—each trajectory may have a different conserved value k, so we must introduce an additional k variable for each trajectory given as training data. p_i is assumed to be a function that returns the index of the trajectory associated with the ith sample.

The solution to this optimization problem can be found in closed form in a manner very similar to the way one might derive principal component analysis [5]. Given u, we can solve for the optimal values of each k_j. Substituting these k values in terms of u, we ultimately arrive at the equivalent optimization problem (defining Q):

$$Q := \sum_i \left(\dot{z}_{[i]} - \mu_{p_i} \|\dot{z}_{[i]}\|^2\right)^T \tag{11}$$

$$\max_{\|u\|=1} u^T Q^T Q u \tag{12}$$

where the μ_j are values obtained by solving for the k_j. The well-known solution to this problem is the normalized eigenvector corresponding to the greatest eigenvalue of $Q^T Q$.

To summarize, in the noise-free case, if $m(x)$ depends only on d variables after performing a linear change of coordinates, we will have $D-d$ conservation laws of the form (8). Each of these will correspond to a zero eigenvalue of $Q^T Q$ in the above optimization. The remaining d eigenvectors will form a basis for the space on which $m(x)$ does depend. We refer to these projections as the Fermat components of the data. Figure 2 shows FCA applied to a toy problem.

C. Numerical estimation of the Lagrangian

If the number of Fermat components is far less than the original dimensionality of the data, finding them can dramatically alleviate the curse of dimensionality associated with learning $m(x)$, since we need only perform regression in a space of much reduced dimensionality. Recalling our assumption that we are given the values of $m(x)$ pointwise along trajectories, the only thing that needs to be done in this step is to regress $m(x)$ from the reduced coordinates.

We have found that a variant of Nadaraya-Watson kernel regression seems to work well for this purpose. A regularization term ensures that the estimate approaches a high value far from the data.

D. Efficient planning in high dimensions

After performing FCA on the training data and estimating $m(x)$ in the reduced space, the only task that remains is to generate new trajectories interpolating given pairs of points. Again, this is generally a computationally intractable path planning problem in high dimensions. However, we can in practice often find a nearly-optimal solution to the path planning problem in the original high-dimensional space by solving a sequence of low-dimensional planning problems. To see why this is so, we revisit the cost functional (1) and examine the consequences of $m(x)$ depending only on a small number of coordinates.

At this point, it will be useful to introduce some terminology. We refer to that subspace upon which $m(x)$ does not depend as the *cyclic subspace*; the remainder of the space is the *acyclic subspace*. From now on, let $p(t)$ be the projection of the trajectory onto the acyclic subspace, and let $q(t)$ be the projection of the trajectory onto the cyclic subspace. We can start by making a simple substitution in (4):

$$J\{x\} = \int_{t_1}^{t_2} \|\dot{x}\| C(p(t)) dt \tag{13}$$

We can also perform a change of variables to arclength of the path, via $ds = \|\dot{x}\| dt$:

$$J\{x\} = \int_{s_1}^{s_2} C(p(s)) ds \tag{14}$$

Our goal now is to turn (14) into a cost functional defined on paths in the acyclic subspace. Unfortunately, it is not quite in that form yet because s is the arclength of the original, high-dimensional path; we would like to change variables from s to s_p, the arc length of $p(t)$.

Let $ds_q = \|\dot{q}\| dt$ be the arclength element of $q(t)$. Since $p(t)$ and $q(t)$ are orthogonal, $ds^2 = ds_p^2 + ds_q^2$; this expression allows us to solve for ds_p/dt in terms of ds/dt and ds_q/dt. For the purpose of computing an optimal path, we note that we can assume that $\|\dot{x}\| = 1$, since the integral is independent of any particular time parameterization. This implies that $ds/dt = 1$, and therefore $ds_p/dt = ds_p/ds$. ds_q/dt can be computed by noting that each of the $D-d$ *cyclic coordinates* obey a conservation law of the form (8). Putting this all together yields

$$\frac{ds_p}{ds} = \sqrt{1 - \sum_{i=1}^{D-d} \left(\frac{k_i}{C(y(s))}\right)^2} \tag{15}$$

Knowing ds_p/ds, and for now assuming that this quantity is strictly positive, we can change variables in (14) from s to s_p. This yields the desired expression:

$$J\{p\} = \int_{s_{p_1}}^{s_{p_2}} \frac{C(y(s_p))}{\sqrt{1 - \sum_{i=1}^{D-d} \left(\frac{k_i}{C(y(s))}\right)^2}} ds_p = J\{x\} \quad (16)$$

The significance of this expression is that in order to compute the cost of a high-dimensional path, we need only be given its projection $p(t)$ onto the Fermat components and the $D-d$ constants k_i associated with the conservation laws in the other dimensions. Therefore, in order to find a minimum-cost path, it is sufficient to search over the space of low-dimensional trajectories and the $D-d$ constants. However, it is possible to do better than this.

Consider the $D-d$ conservation laws of the form (8) that apply to each component q_i of q. If we change variables to s_p, given $p(s_p)$, we can simply integrate both sides of the conservation law to compute $q_i(s_p)$, since $C(x)$ does not depend on q_i. This results in the following expression for the trajectories in the cyclic subspace:

$$q_i(s'_p) - q_i(0) = k_i \int_0^{s'_p} \frac{ds_p}{\sqrt{C(p(s_p))^2 - \sum_{i=1}^{D-d} k_i^2}} \quad (17)$$

Since the integral on the right-hand side is independent of i, this expression allows us to compute any ratio k_i/k_j (for $k_j \neq 0$) from the known start and end points of the curve. Therefore, given any nonzero k_i, we can compute all of the other k values from this expression—the exceptional case $k_j = 0$ occurs if and only if the net distance traveled in q_j is zero. This allows us to finally conclude that to find a minimum-cost path in the original high-dimensional space, it is sufficient to search over the space of low-dimensional trajectories and a single constant k_i, the other k values being fully determined by k_i.

We can formulate this search problem as a one-dimensional root-finding problem to find the correct value of k_i. Each iteration of the root-finding algorithm finds the low-dimensional trajectory that minimizes the cost functional (16) given k_i (and the dependent k values). Given this trajectory, the associated high-dimensional trajectory can then be computed using (17). The value of k_i cannot be correct unless the endpoint of this trajectory is correct. Therefore, the objective of the root-finding algorithm is to find the value of k_i such that the endpoint of the high-dimensional trajectory is correct. Furthermore, it is not difficult to show that the minimum such k_i should be found to ensure that the associated path is of minimum cost.

The computationally intensive part of the planning is obviously the search for the optimal trajectory in the low-dimensional space. This step can be accomplished in $O(d\alpha^d \log \alpha)$ time (for some constant α) using the Fast Marching method [15], which finds an approximate solution to the continuous Eikonal equation [14] on a discretized domain. The Fast Marching method has the same time complexity as Dijkstra's algorithm, and is very similar to it in most other respects.

Finally, we note that this procedure does not necessarily find the globally optimal path in the high dimensional space. This is largely due to the assumption made above that $ds_p/ds > 0$, which is necessary for the change of variables to be valid. If the distance traveled in the cyclic subspace is sufficiently large, the optimal path may require k values large enough to ensure that $ds_p/ds = 0$ for some portion of the path; this corresponds to motion contained completely within the cyclic subspace. In such cases, the algorithm presented above will find an approximate solution with k values potentially smaller than that of the globally optimal solution. Unfortunately, a complete analysis of these issues is outside the scope of this paper; however, we did not find this potential suboptimality to be a major problem in practice.

III. Experiments

We first tested our method by applying it to human motion capture data obtained from the CMU Motion Capture Database. The data consists of the three-dimensional trajectories of 330 markers placed on a human performing a variety of different actions, recorded at 120 Hz. For our experiments, we manually selected a subset of 24 disjoint sequences depicting one of three actions: jumping jacks, side twists, and knee-elbow touches. We held out three of these, one for each action, and trained on the remaining 21 sequences; i.e., we found the top three Fermat components of the data, then learned $m(x)$ in the resulting three-dimensional space with regularized Gaussian kernel regression.

For each of the three held-out examples, we then found a new sequence interpolating the start and end poses by finding a minimum-cost path in the original 990-dimensional space. In each case, our method (on a 3 GHz Intel Xeon processor) found the optimal path in about one minute.

We implemented two other simple methods for comparison—one that takes into account the boundary conditions, but not the training data; and one that takes into account the training data, but not the boundary conditions. The first is simply linear interpolation between the start and end poses. In the second method (henceforth referred to as the "integration method"), we first trained a mapping from poses in the original high-dimensional space to velocities observed at those poses using nonlinear Gaussian kernel regression. Given an initial condition, we then integrated these velocities to obtain a trajectory. Results are shown in Figures 1 and 3.

As these figures attest, only the knee-touch behavior is adequately described by linear interpolation. Linear interpolation on the jumping jack produces an unnatural motion that lifts the hands vertically up, staying close to the body. It also fails to reconstruct the small hop present in the motion, as can be seen by observing the trajectories of the feet. In the side twist, linear interpolation causes the arms to shrink together, meet at the center of the body, and expand out again. The integration method completely fails to produce a plausible trajectory for the knee touch, getting stuck in

an unlikely pose. Its performance is better in the other two cases, although it never attains the final pose, as seen clearly in Figure 3. The individual appendages also appear less coordinated in this method; in the jumping jack, the right arm initially falls slack by the side, and lags behind the left. During the side twist, the arms initially twist out of the desired plane of rotation before rotating about the desired axis. The integration method did seem to capture some initial small details of the knee-touch well, but completely failed to reconstruct the main knee-touching behavior.

The results obtained with FCA address most of these issues. In each case, the trajectories reached the end point with very little error, and the interpolating sequence captured the main features of the motions very well. The arcs traveled by the arms are well-defined, as well as the small jump evident in the feet, though it is not quite as pronounced as in the original sequence. The rigid relative positions of the arms are preserved well in the side twist, and the nearly-linear knee touch motion is also executed well, except for some incidental nonlinear motion of the arms.

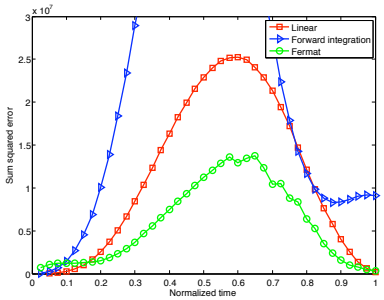
We additionally attempted to use FCA to reconstruct a novel action sequence not present in the original training set. The initial pose given was the initial pose of a jumping jack, and the final pose was the final pose of a side twist. The result is shown in Figure 4. As expected, the resulting sequence resembles half of a jumping jack, followed by a side twist. The sequence appears mostly natural, except that the right arm bends backward in an awkward way upon starting the side twist.

Finally, we applied FCA to data collected from a realistic physical simulation of a humanoid robot based on the Open Dynamics Engine. A hand-coded controller guided the robot through a normal walking gait as we logged the Euclidean coordinates of major joints on the robot. We then partitioned the data at each footfall, training on individual steps made in the course of the gait. We held one of these steps out from the training set and subsequently synthesized the motion between two footfalls based on our learned model via high-dimensional planning.

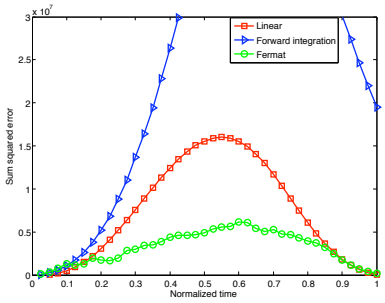
Linear interpolation actually managed to account for much of the overall motion, which would seem to explain the minimal quantitative difference between linear interpolation and our method (Figure 6). However, visualizing the resulting synthesized trajectories shows that linear interpolation failed to reconstruct the parabolic trajectory of the flight leg, resulting in the foot dragging on the ground. Our method was able to successfully capture this important detail by finding a nonlinear trajectory interpolating the start and end poses. The integration method produced results that rapidly diverged from a feasible gait, as seen in Figure 6.

IV. Conclusions

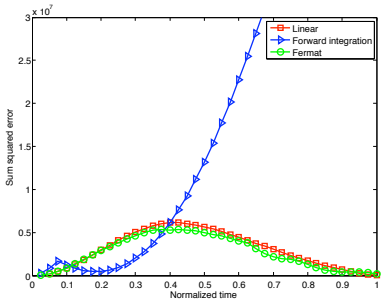
We have presented a two-step approach aimed at solving the high-dimensional learning and planning problems associated with modeling physical trajectory data. First, we find the hidden, low-dimensional structure of a kinetic Lagrangian with FCA; then, we exploit this structure to solve the



(a) Jumping jack

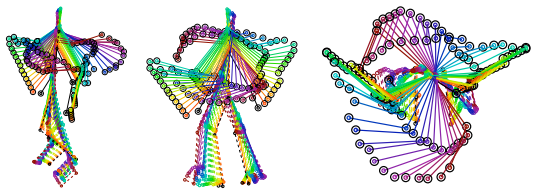


(b) Side twist



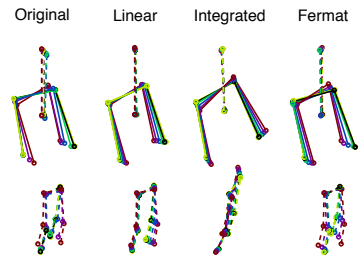
(c) Knee touch

Fig. 3. Sum of squared errors between true pose and corresponding poses in the reconstructed sequence. Boxes mark the trajectory obtained with linear interpolation, triangles mark the trajectory obtained with the nonlinear regression + integration method, and circles mark the trajectory obtained via FCA.

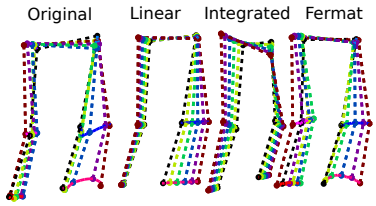


(a) Front (b) Rear (c) Top

Fig. 4. A few views of an experiment reconstructing a novel action sequence. Sequence begins with a jumping jack and ends with a side twist.



(a) Whole-body view



(b) Closer view of legs

Fig. 5. Visualization of experiment reconstructing a portion of a robot gait sequence from starting and end poses. Note especially the trajectories of the left foot.

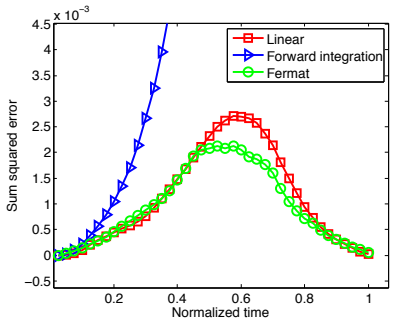


Fig. 6. Quantitative results from the gait reconstruction experiment. Plot shows sum of squared errors between true pose and corresponding poses in the reconstructed sequences.

otherwise intractable planning problem for such a system. We see a few different potential research directions stemming from this work.

Simple physical models with few degrees of freedom have been invaluable in the analysis of the dynamics of complex biological ([16]) and robotic locomotion ([17] [18]). At present, however, these models are usually derived through a combination of inspiration and laborious analysis of models from first principles. Ultimately, it is our hope that FCA or a similar method may eventually evolve into a tool that might aid researchers in this task by identifying low-dimensional views of physical trajectory data that are the most salient for the purpose of predicting physical behaviors.

We also hope that the general idea of identifying structure in cost functions for path planning, and exploiting it to vastly simplify the problem, will stimulate additional research along these lines, as it has proved extremely useful in our specific application.

Finally, there are more short-term desiderata that we would like to address. For instance, a limitation of our method in its current incarnation is that its application is limited to the case where a single linear change of coordinates renders the Lagrangian independent of most of the coordinates. We are currently considering extensions that will weaken this linear assumption. For instance, many additional Lagrangians could be modeled well as being only locally dependent on a few variables. In these cases, we hope that we will be able to find ways to partition space intelligently and apply our method to each partition. We believe that taking such an approach will vastly expand the applicability of our method to real systems.

V. Acknowledgements

We would like to thank the anonymous reviewers for their many insightful comments.

References

[1] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions of Systems Science and Cybernetics*, 1968.

[2] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, 1996.

[3] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

[4] P. Vernaza, M. Likhachev, S. Bhattacharya, S. Chitta, A. Kushleyev, and D. D. Lee, "Search-based planning for a legged robot over rough terrain," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[5] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*. Springer, 2001.

[6] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.-Y. Glorennec, H. Hjalmarsson, and A. Juditsky, "Nonlinear black-box modeling in system identification: a unified overview," *Automatica*, vol. 31, no. 12, pp. 1691 – 1724, 1995, trends in System Identification.

[7] S. Vijayakumar and S. Schaal, "LWPR : An O(n) algorithm for incremental real time learning in high dimensional space," in *Proc. of Seventeenth International Conference on Machine Learning*, 2000.

[8] M. Sekimoto, S. Arimoto, S. Kawamura, and J.-H. Bae, "Skilled-motion plannings of multi-body systems based upon Riemannian distance," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[9] J. Wang, D. Fleet, and A. Hertzmann, "Gaussian process dynamical models," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. Platt, Eds. Cambridge, MA: MIT Press, 2006, pp. 1441–1448.

[10] N. D. Lawrence, "Gaussian process latent variable models for visualisation of high dimensional data," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[11] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*. New York, NY, USA: ACM, 2006, pp. 729–736.

[12] H. Goldstein, *Classical Mechanics*. Addison-Wesley, 1980.

[13] V. I. Arnold, *Mathematical Methods of Classical Mechanics*. Springer-Verlag, 1989.

[14] E. Hecht, *Optics*. Addison Wesley, 2001.

[15] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 93, no. 4, pp. 1591–1595, 1996.

[16] R. Blickhan and R. Full, "Similarity in multilegged locomotion: Bouncing like a monopode," *Journal of Comparative Physiology A*, 1993.

[17] D. E. Koditschek and M. Buhler, "Analysis of a simplified hopping robot," *The International Journal of Robotics Research*, 1991.

[18] T. McGeer, "Passive dynamic walking," *The International Journal of Robotics Research*, 1990.