

Physically-based Motion Models for 3D Tracking: A Convex Formulation

Mathieu Salzmann
TTI-Chicago
salzmann@ttic.edu

Raquel Urtasun
TTI-Chicago
rurtasun@ttic.edu

Abstract

In this paper, we propose a physically-based dynamical model for tracking. Our model relies on Newton’s second law of motion, which governs any real-world dynamical system. As a consequence, it can be generally applied to very different tracking problems. Furthermore, since the equations describing Newton’s second law are simple linear equalities, they can be incorporated in any tracking framework at very little cost. Leveraging this lets us introduce a convex formulation of 3D tracking from monocular images. We demonstrate the strengths of our approach on various types of motion, such as billiards and acrobatics.

1. Introduction

Recovering the 3D motion of dynamical systems from monocular images has been an active area of research for many years. To provide robustness to image noise and occlusions, as well as to overcome the underlying ambiguities of the problem, most practical tracking algorithms exploit motion models. Unfortunately, the existing dynamical models typically suffer from one of the two following shortcomings. Either they are very general, but fail to accurately represent the true physical behavior of the system, or they precisely describe the observed dynamics, but are very specific and/or hard to optimize. In short, there is a lack of general and practical motion models.

The motion of any system is governed by laws of physics. Throughout the years, many attempts at encoding these laws into dynamical models have been proposed [12, 27, 6]. Recent approaches have focused on accurately modeling the behavior of a single type of motion, e.g., jogging or walking [24, 5]. As a consequence, the resulting methods do not generalize to other motions than those they have been designed for. Furthermore, the underlying physical laws encoded by these models are highly nonlinear, thus yielding complex inference problems. Therefore, tracking is performed either by simulation, or by minimizing a non-convex objective function. Both approaches are computationally expensive and may yield suboptimal solutions.

As an alternative to those physically-inspired but impractical motion models, many tracking algorithms rely on

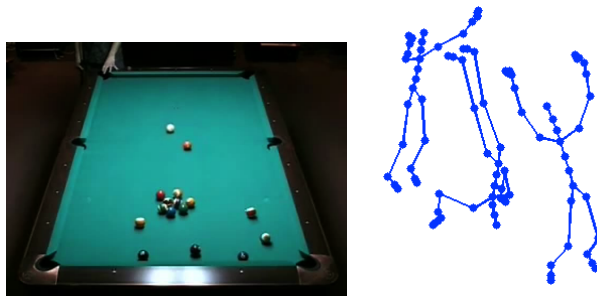


Figure 1. **Billiards shots and acrobatics** are examples of the general tracking scenarios that we address in this paper.

more general Markov dynamical models [19, 7, 11, 8]. In practice, these models are typically used to penalize either overly large displacements (i.e., first-order Markov model), or strong variations of velocity (i.e., second-order Markov model) between consecutive frames. Higher-order models could in principle be employed, but tend to be more sensitive to noise, since they provide weaker motion constraints. Unfortunately, while traditional Markov models are not limited to a specific tracking problem, they often do not reflect the dynamical behavior of real systems.

In this paper, we propose a physics-based dynamical model applicable to general tracking problems. In addition to defining realistic motion constraints, our model has the advantage of providing us with an estimate of the external forces applied to the objects that compose the system. More specifically, we exploit Newton’s second law of motion, which is satisfied by any real-world system. This law can be expressed in terms of linear equalities, and therefore can be incorporated in any tracking system with very little increase in inference complexity. By exploiting this property, we introduce a unified convex formulation of motion tracking for problems as different as bouncing balls, billiards shots, and human activities (e.g., acrobatics).

We demonstrate the effectiveness of our approach on synthetic and real monocular sequences, such as those depicted in Fig. 1, and show that it yields more accurate results than those obtained without motion model, as well as with a traditional first-order Markov model. Furthermore, while the second-order Markov model can be seen as a special case of our model, we show that other settings of our approach often outperform this standard regularizer.

2. Related Work

Tracking moving, and potentially articulated objects in monocular video sequences is a poorly-constrained problem due to, for instance, the presence of noise and occlusions. Existing methods have proposed various solutions to incorporate additional knowledge about the problem of interest to improve 3D reconstruction. While for articulated and non-rigid pose estimation, much effort has been put into designing static constraints [4, 19, 23, 18], here we focus on dynamical models, which are typically used in conjunction with the static ones.

A natural way to represent the motion of a dynamical system is to model the underlying physical laws that govern it. Physics-based motion models take their root in computer graphics [26, 9, 16, 2, 1]. Following a similar motivation, physically-inspired dynamical models were applied to tracking non-rigid motion [12], as well as human activities [27, 6]. Recently, accurate models have been proposed to represent specific activities, such as jogging or walking [24, 6]. While very well-suited for the particular motion they are designed for, these models do not generalize to other activities. More importantly, the above-mentioned physics-based dynamical models typically yield nonlinear constraints. The resulting methods therefore rely either on non-convex optimization, or on analysis-by-synthesis, both of which are computationally expensive and tend to yield suboptimal solutions.

As an alternative to physically-inspired models, statistical learning techniques have been proposed to discover the laws governing the dynamics of a specific system from training data. For human motion tracking, auto-regressive (AR) models have been employed to learn linear relations between consecutive poses [14]. To allow for greater variability in motion, switching models were introduced [10, 15]. Since the AR models are more stable for low-dimensional parameterizations, they were used in conjunction with linear subspace models [3, 21]. Following a similar idea of modeling dynamics in a low-dimensional latent space, the Gaussian Process Dynamical Model [25] was applied to human motion tracking [22], thus allowing for nonlinear temporal relations. While effective, these models do not generalize beyond the problem they have been trained on, and thus cannot be applied to more general tracking scenarios.

Since existing physics-based models yield complex inference problems, and learned models require training data for every possible motion type, many tracking methods utilize the simpler, yet more general Markov dynamical models. First-order [7, 11, 8, 17] and second-order [19, 23] Markov models are the most popular ones. They are used to either bound the range of motion, or limit the variations of velocity between consecutive frames in a video sequence. The main advantage of these models is their applicability to general tracking scenarios. Unfortunately, the constraints they pro-

vide are approximate, and as a consequence their predictions often tend to be suboptimal.

Here, we take advantage of both physics and Markov models. In particular, our approach exploits Newton’s second law of motion to derive a model that generalizes over the second-order Markov model by additionally accounting for the forces applied to the system, thus yielding more flexibility. As a consequence, our physically-based dynamical model is simple enough to yield a convex formulation of tracking, while being valid for any real-world tracking scenario.

3. A Physically-based Motion Model

In this section, we first review some basic concepts of kinematics, from which we then derive our model.

3.1. A Review of Kinematics

Newton’s second law of motion is at the core of our dynamical model. Hence, we first briefly go over its general formulation, as well as the basic concepts it relies on. Let $\mathbf{y}_i \in \mathbb{R}^3$ be the 3D position of the i -th particle belonging to a system of particles. The **linear momentum** of this single particle can be expressed as

$$\mathbf{p}_i = m_i \mathbf{v}_i, \quad (1)$$

where \mathbf{v}_i is the instantaneous velocity of the particle, and m_i is its mass.

Newton’s second law of motion states that the acceleration \mathbf{a}_i produced by a force \mathbf{f}_i applied to a body of mass m_i has the same direction as \mathbf{f}_i , and a magnitude directly proportional to that of \mathbf{f}_i and inversely proportional to m_i , i.e., $\mathbf{f}_i = m_i \mathbf{a}_i$. This can be related to the linear momentum by saying that the instantaneous change in the linear momentum of a particle is equal to the resultant external force acting on that particle. This can be written as

$$\mathbf{f}_i = \frac{d\mathbf{p}_i}{dt} = m_i \mathbf{a}_i, \quad (2)$$

where \mathbf{f}_i is the sum of all external forces applied to the particle. This formulation relies on the fact that the mass is constant, i.e., $\frac{dm_i}{dt} = 0$.

In practice, the resultant external force \mathbf{f}_i can arise from many different types of forces. In our tracking scenarios, we will encounter gravity, friction, collisions, contact and internal forces arising from the action of human muscles.

3.2. Formulation of our Model

Based on the equations describing Newton’s second law of motion, we now derive the formulation of our dynamical model. Following physics-based animation techniques, we approximate the instantaneous velocity at time t using finite

differences. This lets us re-write Eq. 2 as

$$\mathbf{f}_i^t \approx m_i \left(\frac{\mathbf{y}_i^t - 2\mathbf{y}_i^{t-1} + \mathbf{y}_i^{t-2}}{\Delta t} \right). \quad (3)$$

Let us now consider the case of a system of N_p particles undergoing motion for N_f time frames. Let $\mathbf{y}^t = [(\mathbf{y}_1^t)^T, \dots, (\mathbf{y}_{N_p}^t)^T]^T$ be the $3N_p$ dimensional vector containing the 3D locations of all particles at time t , and let $\mathbf{y} = [(\mathbf{y}^1)^T, \dots, (\mathbf{y}^{N_f})^T]^T$ be the $3N_p N_f$ dimensional vector composed of the 3D locations for the whole sequence. By treating all the particles independently, we can impose Newton's second law on each particle. This can be expressed as a set of linear constraints in terms of the particles' coordinates as

$$\mathbf{M}\mathbf{y} = \mathbf{f}_{known} + \mathbf{f}, \quad (4)$$

where \mathbf{f}_{known} specifies the known external forces of the system, e.g., gravity, and $\mathbf{f} \in \mathbb{R}^{3N_p(N_f-2)}$ is the vector of additional unknown forces that act on each particle at each time instant. \mathbf{M} is the $3N_p(N_f-2) \times 3N_p N_f$ matrix that indexes over \mathbf{y} to compute the accelerations of Eq. 3, and takes the form

$$\mathbf{M} = \begin{pmatrix} 1 & & -2 & & 1 & & \\ & \ddots & & \ddots & & \ddots & \\ & & 1 & & -2 & & 1 \end{pmatrix},$$

with all but three elements in each row equal to 0. Note that this matrix has the same form as the matrix encoding a second-order Markov model. However, in such a Markov model, the right-hand side of Eq. 4 is assumed to be 0, which shows both that the second-order Markov model is only valid for very specific types of motion, and that our model is a generalization of it. Note that the mass of the particles has been omitted in Eq. 4. This is to avoid assuming that the mass of each particle is known a priori. As a consequence, forces are expressed per unit of mass. This does not limit the applicability of our model, but only implies that the recovered forces will be up to a scale.

Solving the under-constrained linear system of Eq. 4 for \mathbf{y} and \mathbf{f} yields a physically-valid estimate of the 3D motion and of the forces. In the next section, we show how to incorporate this model into a convex formulation of tracking.

4. A Convex Formulation of 3D Tracking

In this section, we introduce our convex formulation of 3D tracking, which solves for the 3D locations of the particles, as well as for the forces acting on them. Given our physically-based motion model described in Section 3.2, we formulate tracking as the optimization problem

$$\begin{array}{ll} \underset{\mathbf{y}, \mathbf{f}}{\text{minimize}} & \mathcal{L}(\mathbf{y}) + \lambda \mathcal{R}(\mathbf{f}) \\ \text{subject to} & \mathbf{M}\mathbf{y} = \mathbf{f}_{known} + \mathbf{f} \\ & C(\mathbf{y}) = 0 \end{array} \quad (5)$$

which minimizes a problem-dependent loss $\mathcal{L}(\mathbf{y})$ under the motion constraints defined by our dynamical model and problem-specific constraints $C(\mathbf{y}) = 0$. In addition to minimizing $\mathcal{L}(\mathbf{y})$, we introduce a regularization term $\mathcal{R}(\mathbf{f})$ on the forces, whose influence is regulated by λ . In our experiments, we tested the following convex regularizers:

- **Sparsity:** $\mathcal{R}(\mathbf{f}) = \|\mathbf{f}\|_1$. The ℓ_1 -norm encourages the forces to be sparse, thus favoring simple explanations of motion. This regularizer is particularly well-adapted to collisions and instantaneous forces.
- **Small magnitude:** $\mathcal{R}(\mathbf{f}) = \|\mathbf{f}\|_2$. The ℓ_2 -norm encourages forces to remain small. As opposed to the sparsity-inducing regularizer, this is better suited to model phenomena such as friction. In the absence of known forces, this setting is equivalent to a second-order Markov model.
- **Elastic net:** $\mathcal{R}(\mathbf{f}) = \|\mathbf{f}\|_1 + \gamma \|\mathbf{f}\|_2$. This regularizer attempts to combine the benefits of the previous two regularizers. γ sets the relative influence of both terms.
- **Structured sparsity:** $\mathcal{R}(\mathbf{f}) = \|\mathbf{f}\|_{1,2}$. The mixed $\ell_{1,2}$ -norm encourages sparsity over groups of variables. For our purpose, we consider the (x, y, z) -coordinates of each point as a group. This has the advantage over the ℓ_1 -norm of not favoring the non-zero forces to appear along the canonical axes.

Since these different regularizers are all convex, if $\mathcal{L}(\mathbf{y})$ and the constraints $C(\mathbf{y}) = 0$ are convex functions of \mathbf{y} , the problem of Eq. 5 is a convex optimization problem whose global minimum can be obtained using standard solvers [20].

For tracking purposes, the loss $\mathcal{L}(\mathbf{y})$ typically contains, but may not be limited to, an image-based loss $\mathcal{L}_{image}(\mathbf{y})$. For all the different examples that we discuss below, we used the reprojection error of the particles on the image as $\mathcal{L}_{image}(\mathbf{y})$. This loss is similar in spirit to the image term in [22]. Furthermore, it can be expressed as a convex function. Note that any other image loss could be employed. However, non-convex ones would come at the price of losing the overall convexity of the framework.

Let us assume that we are given as input the image locations (u_i^t, v_i^t) of each particle at time t , as well as the matrix \mathbf{A} of internal camera parameters. As shown in [17], the fact that particle i reprojects at the correct location can be written as the system of linear equations

$$\mathbf{P}_i^t \mathbf{y}_i^t = 0, \quad \mathbf{P}_i^t = \mathbf{A}_{2 \times 3} - \begin{bmatrix} u_i^t \\ v_i^t \end{bmatrix} \mathbf{A}_3,$$

with $\mathbf{A}_{2 \times 3}$ the first two rows of \mathbf{A} , and \mathbf{A}_3 the third one. We can derive similar equations for each particle, and group all of them into a single linear system $\mathbf{P}\mathbf{y} = \mathbf{0}$. This lets us define our image-based loss as

$$\mathcal{L}_{image}(\mathbf{y}) = \|\mathbf{P}\mathbf{y}\|_2. \quad (6)$$

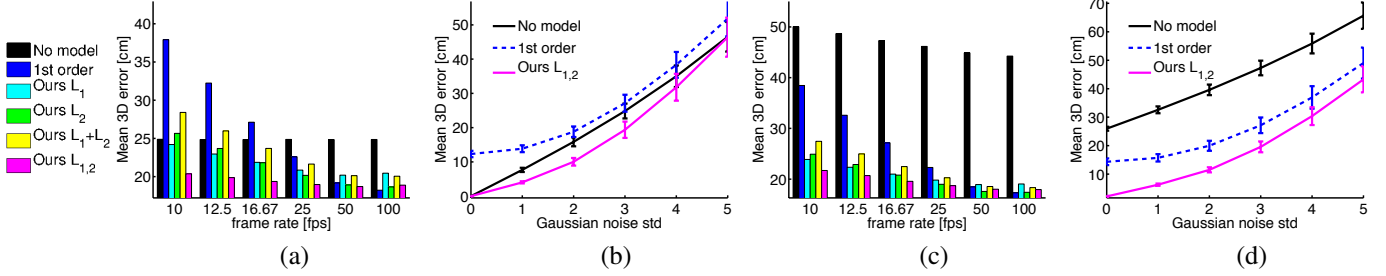


Figure 2. **Tracking bouncing balls.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 3. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 16.6 fps. (c,d) Similar plots as (a,b) for the case where the 2D tracks were lost for up to 0.5 seconds.

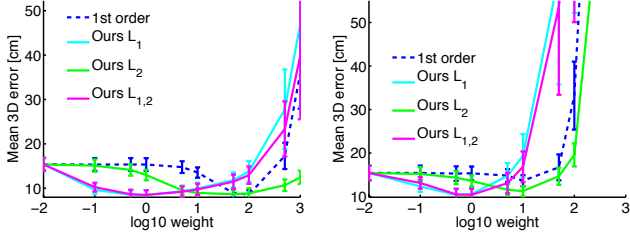


Figure 3. **Influence of model parameters:** Mean error as a function of the motion model weights in log scale. (Left) Framerate = 100 fps. (Right) Framerate = 10 fps. Note that the best weights for our regularizers are stable across different framerates.

In the case of missing data, the rows of \mathbf{P} corresponding to the missing observations can simply be removed.

The formulation in Eq. 5 is very general. We demonstrate its effectiveness in three different tracking scenarios: 3D bouncing balls, billiards and articulated motion. In the remainder of this section, we describe the problem-specific terms $C(\mathbf{y})$ and $\mathcal{L}(\mathbf{y})$. Note that other choices are possible.

4.1. 3D Bouncing Balls

We are interested in recovering the 3D motion of one or more balls bouncing in a room. In this case, gravity plays an important role. Since it has a known and fixed value, we define \mathbf{f}_{known} as the concatenation of $N_p(N_f - 2)$ copies of $\mathbf{f}_{gravity} = m\mathbf{g}$, where $\mathbf{g} = [0, 9.81, 0]^T$ m/s², assuming that the direction of the gravity is in the y -axis of the world coordinate system. Furthermore, unknown forces arise from collisions with the walls of the room, as well as from collisions between the balls. Note that we make no assumptions about the number of walls, or about their location and orientation.

Without additional image information, recovering the 3D position of a single point from its noisy image location is a very ambiguous problem, and a motion model does not suffice to fully constrain it. Therefore, to be more robust, we create additional image correspondences assuming that we are provided with a bounding box around each ball, for example from a 2D tracker. Knowing the diameter of the 3D balls, we can use points sampled on the bounding boxes to create correspondences of the form $\mathbf{P}_{i,j}^t(\mathbf{y}_i^t + \delta_j \mathbf{y}_i^t) =$

0, where j is an index over the sampled points, and $\delta_j \mathbf{y}_i^t$ represents the corresponding known 3D displacement with respect to the center of the ball \mathbf{y}_i^t . Grouping these equations yields an image-based loss of the form $\|\mathbf{P}'\mathbf{y} - \mathbf{q}\|_2$, where \mathbf{q} contains the $\mathbf{P}_{i,j}^t \delta_j \mathbf{y}_i^t$. We can thus re-write the optimization problem of Eq. 5 as

$$\begin{aligned} & \underset{\mathbf{y}, \mathbf{f}}{\text{minimize}} && \|\mathbf{P}'\mathbf{y} - \mathbf{q}\|_2 + \lambda \mathcal{R}(\mathbf{f}) \\ & \text{subject to} && \mathbf{M}\mathbf{y} = \mathbf{f}_{known} + \mathbf{f}. \end{aligned} \quad (7)$$

4.2. Billiards Balls

In this scenario, we have the additional constraint that the balls move on a plane. Here, we assume that we know the normal to the plane, which can be obtained from a few 3D-to-2D correspondences using a PnP method [13]. Given this normal \mathbf{n} , we can enforce the trajectory of each ball to remain planar by satisfying

$$\mathbf{n}^T (\mathbf{y}_i^t - \mathbf{y}_i^{t-1}) = 0. \quad (8)$$

With this constraint only, the balls can still move on parallel planes. To prevent this, we incorporate the additional constraint

$$\mathbf{R}_3 \mathbf{y}_i^1 = \mathbf{R}_3 \mathbf{y}_{i+1}^1 \quad (9)$$

for $N_p - 1$ balls in the first frame, where \mathbf{R}_3 is the third row of the rotation matrix transforming camera coordinates to plane coordinates, typically returned by the PnP method. All these equations can be grouped in a single linear system of the form $\mathbf{B}\mathbf{y} = \mathbf{0}$.

The forces acting on billiards balls in motion are very different from the ones acting on 3D bouncing balls. In particular, gravity has no effect, since it is countered by the contact forces of the balls on the plane. Therefore, there is no known forces for this scenario. On the other hand, we expect to have unknown friction forces in addition to the unknown collision forces. We encode all unknown forces in \mathbf{f} . This lets us re-write the problem of Eq. 5 as

$$\begin{aligned} & \underset{\mathbf{y}, \mathbf{f}}{\text{minimize}} && \|\mathbf{P}'\mathbf{y} - \mathbf{q}\|_2 + \lambda \mathcal{R}(\mathbf{f}) \\ & \text{subject to} && \mathbf{M}\mathbf{y} = \mathbf{f} \\ & && \mathbf{B}\mathbf{y} = \mathbf{0}. \end{aligned} \quad (10)$$

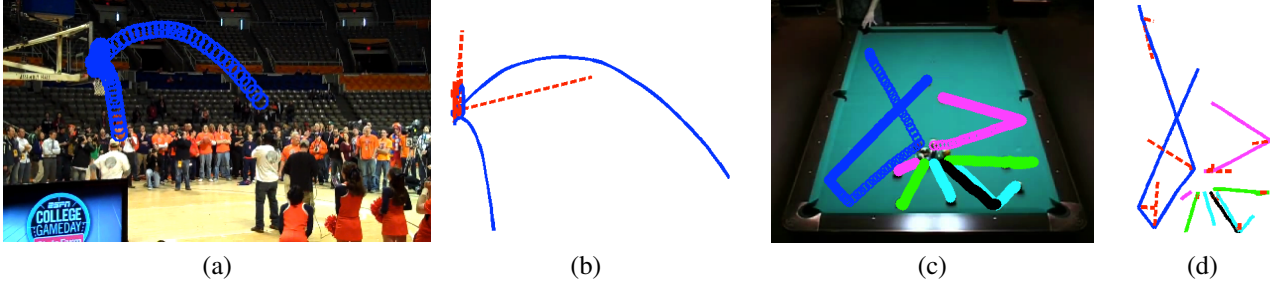


Figure 4. **Tracking balls in real sequences.** (a) Reprojection of the 3D trajectory reconstructed with our approach on the last image of a basketball sequence. (b) Side view of the same trajectory with estimated forces in red. As can be better seen in the video, they match the bounces on the basketball ring. (c,d) Similar plots as (a,b) when tracking several balls in a billiards sequence.

4.3. Articulated Human Motion

Here, we consider the case of human motion tracking and restrict our image-based loss to the one in Eq. 6. Since recovering human pose from such poor image information is very ambiguous, we exploit pose models learned from training data. In particular, we consider both a generative and a discriminative model.

Discriminative case: We rely on Gaussian processes to learn a mapping from image observations to 3D poses from training pairs of images and poses [18]. At inference, we first apply this mapping to get a prediction $\hat{\mathbf{y}}^t$ of the 3D pose at each time t . We then add a term to our loss function $\mathcal{L}(\mathbf{y})$ which encourages the reconstruction to remain close to this prediction. This yields the optimization problem

$$\begin{aligned} & \underset{\mathbf{y}, \mathbf{f}}{\text{minimize}} && \|\mathbf{P}\mathbf{y}\|_2 + \alpha \|\mathbf{y} - \hat{\mathbf{y}}\|_2 + \lambda \mathcal{R}(\mathbf{f}) \\ & \text{subject to} && \mathbf{M}\mathbf{y} = \mathbf{f}_{known} + \mathbf{f}, \end{aligned}$$

where $\hat{\mathbf{y}}$ is the vector of all predictions, and α is a constant.

Generative case: We employ a linear subspace model trained from 3D poses [19]. At inference, we model each pose as $\mathbf{y}^t = \mathbf{y}_0 + \mathbf{U}\mathbf{x}^t$, where \mathbf{y}_0 is the mean pose, and \mathbf{U} is the matrix of eigen-poses. The vector \mathbf{x}^t contains the weights of the linear combination, and is now treated as the unknown of our problem. The vector of all poses can then be defined as $\mathbf{y} = \tilde{\mathbf{y}} + \mathbf{S}\mathbf{x}$, where \mathbf{S} is a block-diagonal matrix containing N_f copies of \mathbf{U} , and $\tilde{\mathbf{y}}$ and \mathbf{x} are the vectors concatenating the mean pose and the weights \mathbf{x}^t , respectively. To prevent these weights from becoming overly large, and thus yield meaningless poses, we add a regularization term of the form $\|\Lambda^{-1/2}\mathbf{x}^t\|_2$, where Λ is a diagonal matrix containing the eigenvalues of the training data covariance matrix. The equations for all frames can be grouped in $\|\mathbf{L}\mathbf{x}\|_2$, where \mathbf{L} is the matrix that concatenates N_f replications of $\Lambda^{-1/2}$. We thus re-write our problem as

$$\begin{aligned} & \underset{\mathbf{x}, \mathbf{f}}{\text{minimize}} && \|\mathbf{P}(\tilde{\mathbf{y}} + \mathbf{S}\mathbf{x})\|_2 + \beta \|\mathbf{L}\mathbf{x}\|_2 + \lambda \mathcal{R}(\mathbf{f}) \\ & \text{subject to} && \mathbf{M}(\tilde{\mathbf{y}} + \mathbf{S}\mathbf{x}) = \mathbf{f}_{known} + \mathbf{f}. \end{aligned}$$

where β is a scalar.

For both the discriminative and generative cases, depending of the motion of interest, \mathbf{f}_{known} may or may not contain gravity. As shown in our experiments, this has very little influence on our results. Note that we do not model the ground plane, which could give us additional constraints.

5. Experimental Evaluation

We now present our results for the different types of tracking problems described in the previous section. We compare our reconstructions with those obtained without using any motion model, as well as by employing a standard first-order Markov model. To this end, we replaced our dynamical model in the optimization problems described in Section 4 with a soft penalty on frame-to-frame displacements. In addition to different framerates, we ran experiments for different levels of additive Gaussian noise in the image measurements. The weights for the Markov model regularizer and for our force regularizers were tuned at the highest framerate and kept unchanged for the rest of the experiment. We measure the 3D error in terms of mean point-to-point distance to the ground-truth, averaged over 10 runs.

5.1. 3D Bouncing Balls

To obtain ground-truth data, we implemented a physics-based simulator and computed 10 different sequences, each seen with a different camera. The bouncing balls are subject to gravity and undergo collisions. We model collision as an instantaneous force that results in a loss of velocity. This is usually described in terms of the coefficient of restitution C_R , which relates the velocities before and after the shock. This coefficient varies between $C_R = 1$ (i.e. perfectly elastic collision) and $C_R = 0$ (i.e. perfectly inelastic collision, where the bodies stick together after the shock). We use $C_R = 0.9$ for our simulations.

Fig. 2(a,b) depicts the error when tracking 10 bouncing balls as a function of the framerate for a noise of standard deviation 3, and as a function of the noise for a framerate of 16.67 fps, respectively. The trajectories for these results were obtained by solving the problem of Eq. 7. Note that our approach outperforms the Markov models. As expected, this is particularly true for low framerates, where they are

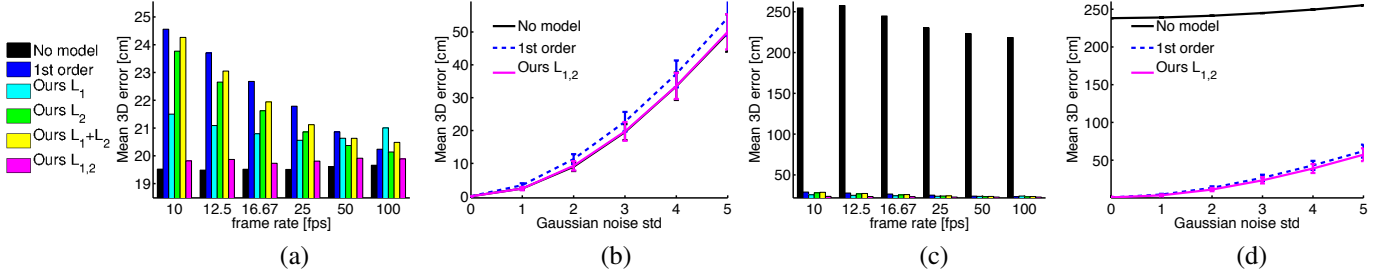


Figure 5. **Multiple balls on a plane.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 3. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 16.6 fps. (c,d) Similar plots as (a,b) for the case where the 2D tracks were lost for up to 0.5 seconds.

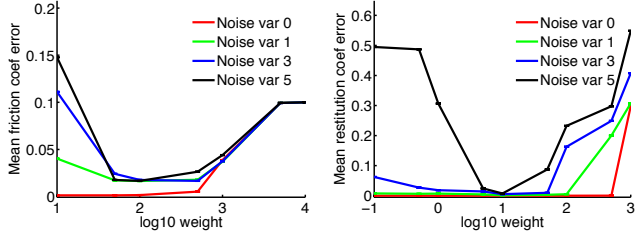


Figure 6. **Recovering physical quantities.** (Left) Mean error as a function of the ℓ_1 force penalty weight when recovering the friction coefficient. The true coefficient was 0.1. (Right) Similar error for the coefficient of restitution. The true coefficient was 0.9.

the least accurate. We simulated missing data by removing random subsets of consecutive frames for each ball, up to 0.5 seconds at a time. Fig. 2(c,d) shows similar plots as before for this case. Note that our method remains accurate, while without a motion model the results degrade. The $\ell_{1,2}$ norm yields the most accurate results among the different regularizers, thus showing that Markov models are suboptimal here.

Fig. 3 depicts the error as a function of the regularization weights for the two extreme framerates used in our experiments (i.e., 10 and 100 fps). Note that the best weights for our regularizers remain more stable across different framerates than for the first-order Markov model. This suggests that our model is less sensitive to this parameter, and hence easier to apply in real scenarios.

We also tracked a basketball shot in a YouTube video. The 2D tracks and bounding boxes were obtained by a simple template-matching method that maximizes the normalized cross-correlation between a template defined in the first image and the following images of the sequence. The re-projection on an input image of the 3D trajectory reconstructed using our approach with the ℓ_1 regularizer is shown in Fig. 4(a), and its side view in blue in Fig. 4(b). Note that the estimated forces, depicted in red, occur when the ball touches the ring, and thus correspond to those we expect.

5.2. Billiards Balls

We implemented a simulator to generate balls moving on a plane, thus mimicking billiards. We employed the same collision model as before and simulated friction as $\mathbf{f}_{friction} = -\mu \|\mathbf{f}_n\| \mathbf{v} / \|\mathbf{v}\|$, where μ is the friction coefficient,

and \mathbf{f}_n is the normal force. In contrast to the bouncing balls, we do not model gravity. We computed 10 different sequences, seen from different viewpoints, and simulated missing data as before. 3D tracking was performed by solving the optimization problem of Eq. 10. Fig. 5 depicts the errors obtained for the case of 5 moving balls colliding with each other and with the table edges. Note that the $\ell_{1,2}$ norm yields more accurate results than the Markov model and the other regularizers. With missing data, it also significantly outperforms the results obtained without motion model.

Fig. 4(c,d) depicts our tracking results on a billiards video from YouTube. The 2D tracks were obtained in a similar manner as in the basketball case. The top view in Fig. 4(d) shows that the trajectory recovered by our approach with an ℓ_1 regularizer matches what we observe in the images, and that the estimated forces correspond to those we expect.

Our model can also be used to estimate physical constants, such as the friction and restitution coefficients, from the reconstructed forces and trajectories. For the friction coefficient, we post-processed the estimated forces by discarding the large ones. We then recovered μ by least-squares fitting based on the equation of friction, using gravity as the normal force. As shown in Fig. 6(left), this results in accurate estimates. Fig. 6(right) depicts the error when estimating the restitution coefficient. In this case, C_R was obtained from the velocities before and after the collisions, which were detected by finding the large forces.

5.3. Human Motion

To address the problem of tracking people performing different activities, we exploited the publicly available CMU motion capture dataset¹ and tested our approach on three different activities: jumping, jogging and acrobatics (i.e., a front handspring). The 2D tracks were obtained by projecting the 3D data using a known camera. Gaussian noise was added to these image locations with standard deviation ranging from 0 to 10. As mentioned in Section 4, since the problem of human body tracking from 2D correspondences is too ambiguous, we augmented our motion model, as well as the baselines, with pose models. We used the poses from a subset of the data as training examples, and the remaining

¹<http://mocap.cs.cmu.edu/>

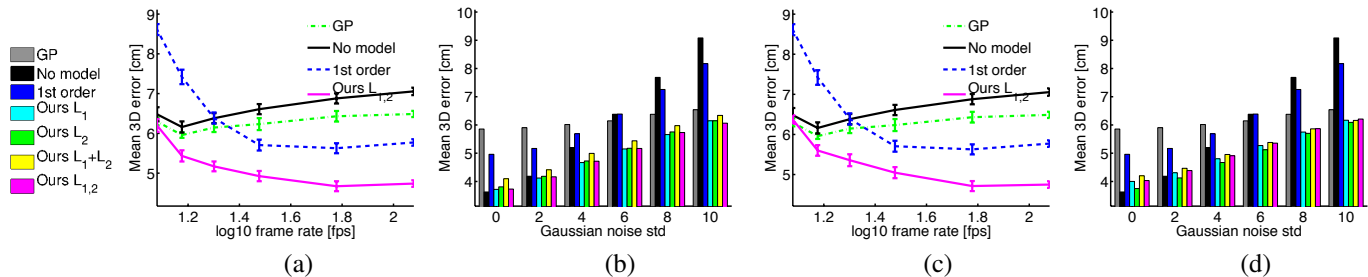


Figure 7. **Tracking a human jumping with a discriminative model.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 6. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 20 fps. (c,d) Similar plots as (a,b) when the gravity is not explicitly encoded in the known forces. Note that this does not significantly affect our results.

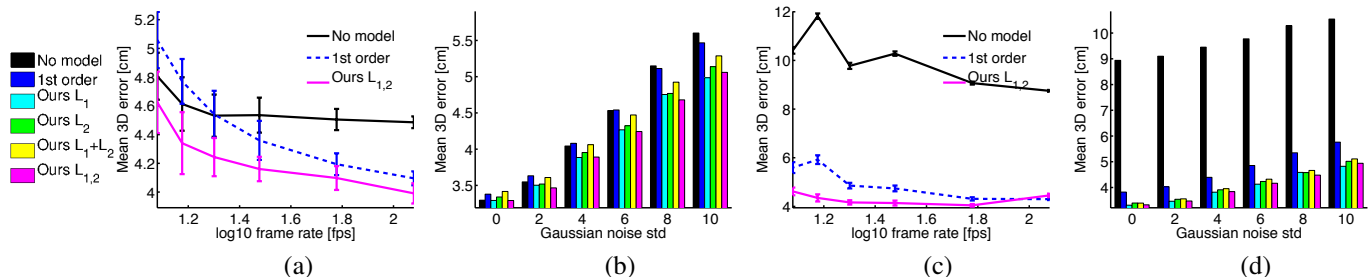


Figure 8. **Tracking a human jumping with a linear subspace model.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 6 and with missing 2D tracks. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 20 fps and with missing 2D tracks. (c,d) Similar plots as (a,b) in the case of full occlusions.

sequences as test data. To simulate missing data (i.e., lost tracks), we removed correspondences in consecutive frames. We also simulated full occlusions by removing all the tracks simultaneously for several frames.

Figs. 7 and 8 summarize our results for a jumping motion when using a Gaussian process mapping and a linear subspace model, respectively. Fig. 7(a,b) shows the 3D errors as a function of the framerate for a Gaussian noise of standard deviation 6, and as a function of the noise for a framerate of 20 fps. Fig. 7(c,d) shows similar plots when the gravity was not explicitly encoded in the known forces. Note that this has little influence on our results. Fig. 8 depicts the errors in the presence of missing 2D measurements and full occlusions. As can be observed from the plots, our approach is robust to these phenomena. In Fig. 9, we show the errors obtained for a jogging motion with a Gaussian process mapping. As before, the explicit use of gravity has little influence on our results. Fig. 10 depicts the errors obtained for the front handspring motion when relying on a linear subspace model. Similarly as for the jumping motion, missing data and full occlusions do not significantly affect our reconstructions. Note that, in all these scenarios, our model consistently outperforms the first-order Markov model. Furthermore, in most experiments, the $\ell_{1,2}$ norm gives the best performance, with the ℓ_2 norm being second best. This shows that other settings of our approach often outperform the traditional special case of second-order Markov models. Finally, in Figs. 1 and 11, we show examples of reconstructions obtained with an image noise of standard deviation 4.

6. Conclusion and Future Work

In this paper, we have presented a physics-based dynamical model that combines the advantages of being generally applicable and easy to optimize. By exploiting our model, we have introduced a convex formulation of 3D tracking that not only recovers 3D trajectories, but also yields an estimate of the forces applied to the system. Finally, we have shown that our approach is general enough to handle very different tracking scenarios, and that it outperforms the standard first- and second-order Markov models, the latter being a special case of our model. While we have considered the problem of 3D motion tracking, our dynamical model could also be applied to other problems, such as camera motion estimation, or post-processing of motion capture data. In the future, we plan to study other physical constraints, such as balance, to improve 3D reconstruction.

References

- [1] D. Baraff. Analytical methods for dynamic simulation of non-penetrating rigid bodies. *ACM SIGGRAPH*, 1989.
- [2] R. Barzel and A. H. Barr. A modeling system based on dynamic constraints. *ACM SIGGRAPH*, 1988.
- [3] A. Bissacco. Modeling and Learning Contact Dynamics in Human Motion. *CVPR*, 2005.
- [4] V. Blanz and T. Vetter. A Morphable Model for the Synthesis of 3D Faces. *ACM SIGGRAPH*, 1999.
- [5] M. Brubaker, D. Fleet, and A. Hertzmann. Physics-based Person Tracking Using the Anthropomorphic Walker. *IJCV*, 87(1), 2010.

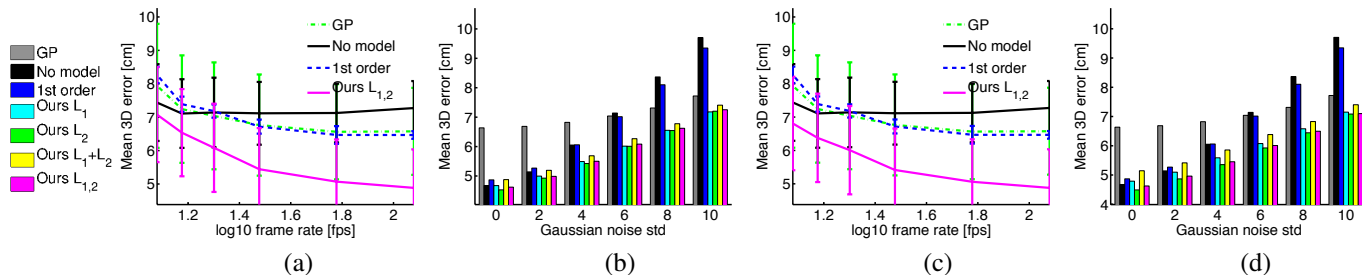


Figure 9. **Tracking a human jogging.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 6. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 20 fps. (c,d) Similar plots as (a,b) when the gravity is not encoded in the known forces. Note that this does not significantly affect the results of our approach.

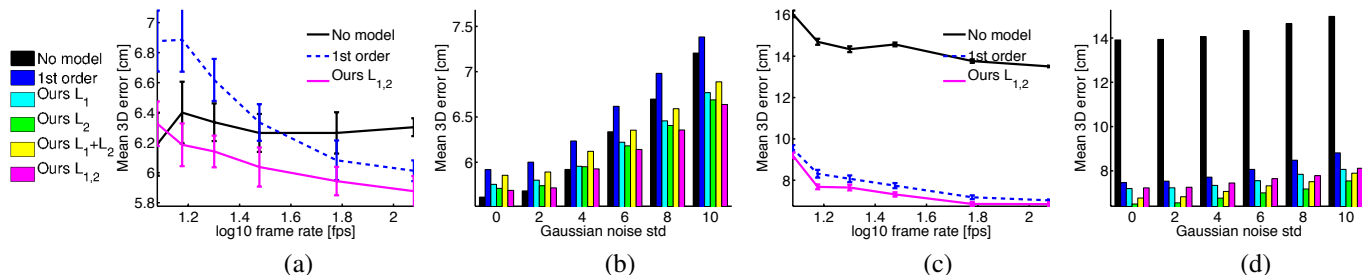


Figure 10. **Tracking a human performing a front handspring.** (a) Mean 3D error as a function of the framerate for an image Gaussian noise of standard deviation 6 and with missing 2D tracks. (b) 3D error as a function of the Gaussian noise standard deviation for a framerate of 20 fps and with missing 2D tracks. (c,d) Similar plots as (a,b) in the case of full occlusions.

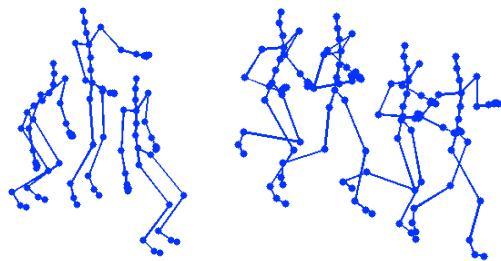


Figure 11. **Tracking human motion.** Reconstructions of a jumping and a jogging sequence with an image noise of variance 4.

[6] M. Brubaker, L. Sigal, and D. Fleet. Estimating Contact Dynamics. *ICCV*, 2009.

[7] K. Choo and D. Fleet. People Tracking Using Hybrid Monte Carlo Filtering. *ICCV*, 2001.

[8] J. Deutscher and I. Reid. Articulated Body Motion Capture by Stochastic Search. *IJCV*, 61(2), 2004.

[9] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating Human Athletics. *ACM SIGGRAPH*, 1995.

[10] M. Isard and A. Blake. A mixed-state Condensation tracker with automatic model-switching. *ICCV*, 1998.

[11] A. Jepson, D. J. Fleet, and T. El-Maraghi. Robust On-Line Appearance Models for Vision Tracking. *PAMI*, 25(10), 2003.

[12] D. Metaxas and D. Terzopoulos. Shape and Nonrigid Motion Estimation Through Physics-Based Synthesis. *PAMI*, 15(6), 1993.

[13] F. Moreno-Noguer, V. Lepetit, and P. Fua. Accurate Non-Iterative $o(n)$ Solution to the Pnp Problem. *ICCV*, 2007.

[14] B. North, A. Blake, M. Isard, and J. Rittscher. Learning and classification of complex dynamics. *PAMI*, 25(9), 2000.

[15] V. Pavlovic, J. Rehg, and J. Maccormick. Learning switching linear models of human motion. *NIPS*, 2000.

[16] Z. Popovic and A. Witkin. Physically Based Motion Transformation. *ACM SIGGRAPH*, 1999.

[17] M. Salzmann, V. Lepetit, and P. Fua. Deformable Surface Tracking Ambiguities. *CVPR*, 2007.

[18] M. Salzmann and R. Urtasun. Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. *CVPR*, 2010.

[19] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic Tracking of 3D Human Figures Using 2D Image Motion. *ECCV*, 2000.

[20] J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones, 1999.

[21] L. Torresani, A. Hertzmann, and C. Bregler. Nonrigid Structure-From-Motion: Estimating Shape and Motion With Hierarchical Priors. *PAMI*, 30(5), 2008.

[22] R. Urtasun, D. Fleet, and P. Fua. 3D People Tracking With Gaussian Process Dynamical Models. *CVPR*, 2006.

[23] R. Urtasun, D. Fleet, A. Hertzman, and P. Fua. Priors for People Tracking from Small Training Sets. *ICCV*, 2005.

[24] M. Vondrak, L. Sigal, and O. C. Jenkins. Physical Simulation for Probabilistic Motion Tracking. *CVPR*, 2008.

[25] J. Wang, D. Fleet, and A. Hertzmann. Gaussian Process Dynamical Models for Human Motion. *PAMI*, 30(2), 2008.

[26] A. Witkin and M. Kass. Spacetime Constraints. *ACM SIGGRAPH*, 1988.

[27] C. Wren and A. Pentland. Dynamic Models of Human Motion. *FG*, 1998.