2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Motion estimation using physical simulation

Damien Jade Duff, *Student Member, IEEE*, Jeremy Wyatt, and Rustam Stolkin

***Abstract*— We consider the task of monocular visual motion estimation from video image sequences. We hypothesise that performance on the task can be improved by incorporating an understanding of physically likely and feasible object dynamics. We test this hypothesis by incorporating a physical simulator into a least-squares estimation procedure. We initialise a full trajectory estimate using RANSAC followed by gradient descent refinement. We present results for 2D image sequences consisting of single ambiguous, visible or occluded balls, as well as results for 3D computer-generated sequences of objects in free-flight with added noise. Results suggest that restricting the estimation to allow only motions that are feasible according to the physics simulator can produce marked improvement when the observed object motion is within the limits of the physics simulator and its world model. Conversely, merely penalising deviations from feasible physical dynamics produces a consistent but incremental improvement over more common dynamics models.**

I. Introduction

In the field of cognitive robotics, the problem of vision is intricately tangled with other cognitive problems, such as control, interaction and manipulation. Practical solutions to these problems arguably need to cut away from a primarily image-centered approach and need to use frameworks suited to a broader class of problems.

In our attempt to broaden the approach we tackle the classic motion estimation problem, but expand on it incrementally. Specifically, we build software that attempts to estimate dynamic parameters of moving objects from image sequences, in addition to pose, kinematic or image parameters. This software also takes into account the impact of physical dynamics on estimated parameters by making use of physical simulation. We make the prediction that incorporating physical simulation in the motion estimation procedure will improve the accuracy of motion estimation with videos consisting of long sequences of images, particularly in the presence of occlusion, distractors, poor image quality or multiple objects.

In the remainder of this section we discuss some related work and describe our own approach. The following two sections discuss motion estimation and physical dynamics in detail, as applicable to this work. Finally we present some results on 3D motion estimation of rigid objects from simulated sequences of noisy poses and 2D motion estimation of a single ball from colour histograms in real-world videos.

All authors are at the School of Computer Science, University of Birmingham, Edgbaston, Birmingham, B15 2TT, United Kingdom.
{D.J.Duff,J.L.Wyatt,R.Stolkin}@cs.bham.ac.uk

A. Further Motivation

While our approach is primarily aimed as an iteration on conventional solutions in robotics, the direction that we are driving in has some justification in the abilities that humans display. Specifically, in humans non-visual information is important in visual tasks; very early in life objects start to hold implications outside the visual and sensory array. If we are to approximate these human capabilities we need to find ways of making judgements about objects and their location that make use of expectations about the way that objects behave.

More practically, good quality motion estimation and tracking of single or multiple objects in the presence of occlusion, distractors, glare, and blur is important in a number of areas. Robotic manipulation is one such area, where a robot's interaction with passive objects can lead those objects to move in complicated ways, tumbling or falling for instance. An understanding of how objects move has potential to improve the robot's estimation of object pose. Moreover, there are synergies to be had between control, planning, and visual estimation [1]. The kind of physics-based motion estimation we are investigating will also find application in trajectory estimation in analysis of sports footage, object tracking in virtual reality, and motion capture.

B. Related Work

Work is being done elsewhere on employing 3D physical dynamics while tracking human movements [2], [3], [4]. There, improvement obtained from dynamics models trained from motion-capture data eclipses that obtained from using a priori physics models. Our work differs from that work primarily in that we are tackling the passive object problem, focusing explicitly on scenarios in which objects display non-intentional rather than active/intentional behaviour.

The computer graphics community is interested in a number of related problems – motion capture and motion synthesis being the two most related (e.g. Popovic et al. 2000 [5]). Motion estimation from image sequences has applications in motion capture, and potentially uses a similar framework to constrained motion synthesis. A recent example of this kind of synthesis is the work of Bhat et al., 2002 [6] who estimate the motion of free-flight objects from silhouettes.

There is also a large literature on general motion estimation in long image sequences [7], [8], [9], often posed simultaneously with the problem of structure estimation. Of particular relevance is work that attempts to track objects from image sequences by considering them as deformable 3D objects undergoing local forces calculated from images [10], and earlier work that uses motion capture information to

guide a simulation of arbitrary deformable 3D objects [11]. In our work we consider the motion estimation rather than tracking problem, and we fit to image sequences rather than motion capture data.

C. Approach

Within this broader project, in order to motivate our work, we make the specific prediction that it is possible to use prior knowledge about physical dynamics to improve vision tasks within existing task domains, particularly in motion estimation, by exploiting restrictions on the range of feasible motions and by giving weight to more feasible motions. Indeed, it seems sensible to suggest that a strong and accurate physics model will allow extrapolation and interpolation where data from observations is missing or misleading.

We first test this on computer generated data by automatically generating noisy three-dimensional trajectories of rigid objects in free-flight, and of bouncing. We use a least-squares estimation procedure that incorporates knowledge of free-flight and simple collision dynamics to reconstruct the generated trajectory from the input set of noisy poses.

Secondly, we evaluate the use of physical dynamics on real two-dimensional image sequences of a ball moving through a scene or stationary in it. We use a colour histogram matching procedure to generate ball location hypotheses at each frame and best-fit a trajectory to these hypotheses, again using a least-squares cost function.

Before we present the results of these experiments we discuss our estimation framework and inclusion of physics in more detail.

II. Motion Estimation

The problem of visual motion estimation is related to, but not the same as, the problem of visual tracking. Visual tracking is the problem of maintaining an estimate of instantaneous object location over time - this location can be hand-initialised or the tracker can try and try to obtain its own lock. On the other hand, visual motion estimation is the problem of reconstructing a whole trajectory from an image sequence. Here we tackle the motion estimation problem, because its statement is simpler and it allows us to easily incorporate object dynamics in an explicit fashion.

Clearly, the problems are related and solutions to one can be adapted as solutions to the other. In particular, a naive application of a motion estimation solution to an object tracking problem would entail a growing problem size over time. Object tracking solutions like mean-shift and recursive filtering have the potential advantage that they only need to store information about the current object state. However, a motion estimation solution could be straightforwardly adapted using moving horizon estimation techniques. Conversely, when applied to the motion estimation task, object tracking solutions can suffer from a sensitivity to conditions at the start of the time sequence, to the extent that track is often never obtained.

In object tracking, object dynamics are often incorporated implicitly; for instance by making the assumption that an object will not move far between frames, as with mean-shift and related object trackers where the search for the object in each image frame is initialised from the best location(s) found in the previous frame [12], [13], [14], [15]. In this paper we call this assumption, when made explicitly, "constant displacement dynamics" and use it as a base-case that we compare our new physics-based methods to. Our other base-case is the "constant velocity dynamics" in which the velocity of objects is assumed not to deviate significantly from frame to frame. Both dynamics are usable as a part of any recursive estimation procedure.

We can assign a sum of squares cost to any trajectory, based on the chosen forward dynamics. In the case of translational dynamics we have:

$$\begin{gathered} dyncost(\{<T_t, T'_t>\}) = \\ K_1 \sum_{t=t_i}^{t_f} \left\| dyn_d(T_t, T'_t) - T_{t+1} \right\|^2 + \\ K_2 \sum_{t=t_i}^{t_f} \left\| dyn_v(T_t, T'_t) - T'_{t+1} \right\|^2 \end{gathered}$$

In the above equation $dyncost$ represents the sum of squares error in dynamics over a whole trajectory, $\{< T_t, T'_t >\}$ (T_t being the pose coordinates of the object at time t and T'_t the first derivative of this - i.e. the velocity). dyn_d and dyn_v are functions that represent the forward dynamics of an object, by determining, in the first case, a predicted displacement at timestep $t+1$ given the object state at time t, and, in the second case, a predicted velocity. Rotational dynamics can be specified in a similar way. Weights K_1 and K_2 determine the relative cost contribution of velocity and displacement error.

The constant displacement forward dynamics therefore has $dyn_d(T_t, T'_t) = T_t$ and $K_2 = 0$. i.e. the displacement is not expected to change, and the velocity is not considered.

Similarly, a constant velocity forward dynamics (assuming that each time step is one unit of time) has $dyn_d(T_t, T'_t) = T_t + T'_t$ and $dyn_v(T_t, T'_t) = T'_t$.

Given this framework, any forward dynamics can be incorporated.

The dynamics cost is mirrored by the observation cost, which is, again, a sum of squares cost based on the deviation between predicted observations and actual observations:

$$obscost(\{< T_t, T'_t >\}, \{O_t\}) = \sum_{t=t_i}^{t_f} w_t \left\| obs(T_t) - O_t \right\|^2$$

The obs function, given above, is any function that predicts an observation from an object's location, and O_t is the observation at time t. In the case of our experiments below, obs simply provides the location of an object given its estimated state. $\{w_t\}$ are observation weights.

Given a cost function made up of the weighted sum of observation and dynamics costs, we can attempt to find a trajectory $\{T_t, T'_t\}$ that minimises it - i.e. we attempt to find the least squares in the cost terms. This can be carried out by any general optimisation procedure. We employ the RANSAC algorithm [16] to select inliers and initialise solutions, and we use gradient descent refinement with finite differences to subsequently refine the solution.

The RANSAC procedure relies on an accurate dynamics model to instantiate full trajectories from small randomly sampled subsets of available observations. These trajectories are used to select inliers and initialise the refinement computation. Without an accurate dynamics model, it is necessary either to sample large observation sets (somewhat defeating the purpose of RANSAC) or to frequently risk failing to find a sufficiently accurate trajectory. The constant displacement and constant velocity dynamics models described above do need to make this trade-off.

III. Incorporating Physical Dynamics

Within the above cost-minimisation framework it is a simple extension to add more sophisticated physical dynamics by providing a more sophisticated dyn function. We adapt the commercial PhysX physics engine to this purpose, by supplying it with a rough model of the environment and of the object to be estimated. We call this dynamics a "locally parametrised collision dynamics". It is called "local" because a dynamics cost is calculated locally between pairs of time-points and summed together to provide a trajectory cost, as discussed above.

We also implement a slightly different "globally parametrised collision dynamics". If we allow that no deviation in the trajectory from the dynamics model is possible then we may parametrise an object's entire trajectory in terms of the instantaneous motion parameters of the object at only one time-point during that trajectory, since the dynamics model deterministically specifies the motion parameters of the remaining time-points. In this case, the cost function to be minimised is simply the observation cost $obscost$, and the parameters to be estimated are the velocity and pose parameters at a single point during the trajectory. We call this approach a "global" model because the model allows the entire trajectory to be parametrised in terms of the motion parameters of a single time-point.

IV. Experiments

A. 3D Rigid Object Motion Estimation in Simulation

Our first experiment is intended to determine the workability of the proposed method.

1) Method: We assume in this experiment that the algorithm takes as input 3D pose estimates at each time step, as supplied by some arbitrary pose-estimation routine. We test our algorithm on trajectories consisting of sequences of poses generated by first simulating a trajectory and then adding some noise. Noise is added from a normal distribution with translation and rotation spread parameters σ_t and σ_r. The latter is approximated from the former by observing that a rotation is made up of a set of translations. Noise is also added as outliers parametrised by the outlier rate (the probability of any pose being an outlier, p) and sampled from a uniform distribution. We generate trajectories consisting of rigid objects in free-flight, under the influence of gravity, or bouncing in a simple fashion. A generated trajectory, and the same trajectory with added noise can be seen in fig. 1(a-b). This experiment was implemented in Octave/Matlab and the simple physics models used were programmed by hand. All collisions were assumed elastic and the object was considered a frictionless perfect sphere. A consequence of these assumptions is that rotation occurs effectively independently of translation. See table I(a) for object physical parameters.

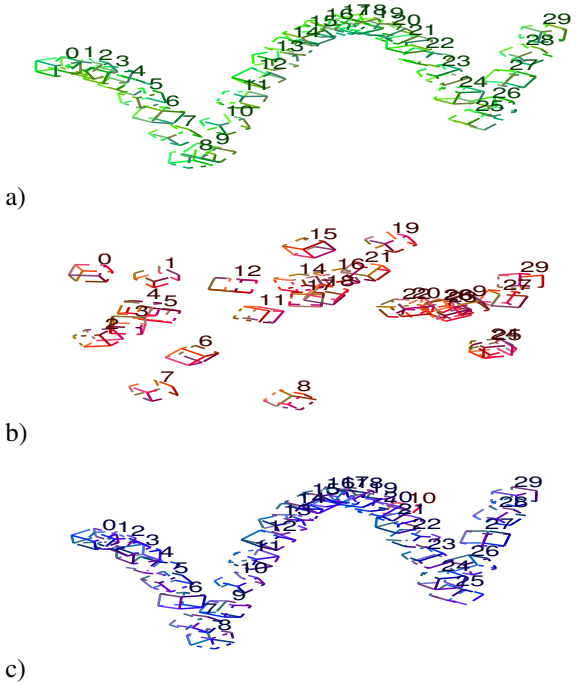

Fig. 1. A single run of the 3D free-flight trajectory estimation algorithm (RANSAC + refinement) on a bouncing rigid object. (a) Green: The veridical (simulated) trajectory. (b) Red: The same trajectory with added normally distributed noise ($\sigma_t = 50cm$) and outliers ($p = 0.1$). (c) Blue: The trajectory fit to it using RANSAC and refinement.

In general, when calculating the distance between a predicted pose and an observed pose, we need some way of comparing poses that takes into account both translational and rotational error. We motivate this calculation by making the assumption that we are interested in the integral sum of squared error distance across all points in the object. By integrating analytically, we obtain the pose error cost $|V|t_e^2 + 2(1 - cos\theta_e)\omega_e^T I \omega_e$, where t_e is the translational error, θ_e, ω_e comprise the rotational error in angle-axis form, $|V|$ is the volume, and I is the inertial matrix of the object, calculated assuming a constant unit density. Dividing by the volume $|V|$ and taking the square root obtains a scale-free error in the original units.

In this experiment, the dynamics model used in estimation is the same as that used to generate data initially. As a result, we only need to make use of the "globally parametrised dynamics model" mentioned above, since we know that the dynamics model used is sufficient to model the data. In the refinement phase we employ the Levenberg-Marquadt procedure [17] to minimise the sum of squared errors, initialised with a random guess. See table I(b-c) for the parameters of the RANSAC and refinement algorithms.

2) Results: The output of a sample estimation run can be seen in fig. 1(c), where output of the estimation algorithm

is compared against the noisy data that is input into the algorithm (b), and is also compared to the ground-truth trajectory before noise was added (c).

Fig. 2 shows the result of applying the estimation algorithm in the presence of varying amounts of noise, with or without a ground plane to induce bouncing and with or without the RANSAC initialisation and data selection step.

With respect to estimating the translational component of motion, these results show that the refinement algorithm fares worse in the presence of bounces, though RANSAC is on the whole resistant to the presence of bouncing, and is robust to higher levels of outlier noise, which is expected. However, with increasing normally distributed noise, the performance of RANSAC becomes worse than that of refinement. Further, we see that the presence of bounces and increasing normal noise interact to worsen the performance of the refinement algorithm, and the presence of bounces together with increasing outlier rate interact to worsen the performance of the RANSAC algorithm.

It can be seen that lengthening the sequence from 2 to 14 (hence increasing the number of rotations) results in a higher rotational cost per-step for the refinement algorithm, a trend reversed by the use of RANSAC. For even longer sequences, the refinement algorithm seems able to benefit from the larger data set.

3) Discussion: As well as being able to deal better with long-tailed noise, it does seem that RANSAC is able to mitigate the presumed local cost-minima introduced by rotations and by bouncing balls. However, its faster deteriorating performance in the presence of increasing normal noise is to be expected given that RANSAC only passes a subset of data points (the inliers) on to the refinement calculation, not taking full advantage of the properties of normal noise.

The interaction between the presence of bounces and noise suggests that noise exacerbates the effect of local minima. Indeed, in the presence of an infinite number of observations and no noise, simple bounces can induce no local minima in the cost function.

We note that the assumption of the existence of a pose-estimation routine that can provide 3D pose estimates is clearly not universally valid, though it does allow us to construct our algorithm in a feature-agnostic way. More generally, the observation cost function can in theory be based on any arbitrary least-squares likelihood cost.

B. 2D Ball Motion Estimation from Real Images

Having ascertained the practicability of the proposed method, it is necessary to apply it to a real-world problem. We apply it to the problem of tracking a ball as it moves through a scene in a small number of video image sequences, some frames of which can be seen in fig. 3.

1) Method: We model the ball's appearance using a normalised colour histogram, and for each video frame we determine the points in the image that correspond to areas that best match the ball according to the Bhattacharyya distance measure. This measure is in wide use in the literature for tracking from colour histograms [12], [13], [14], [15]. We

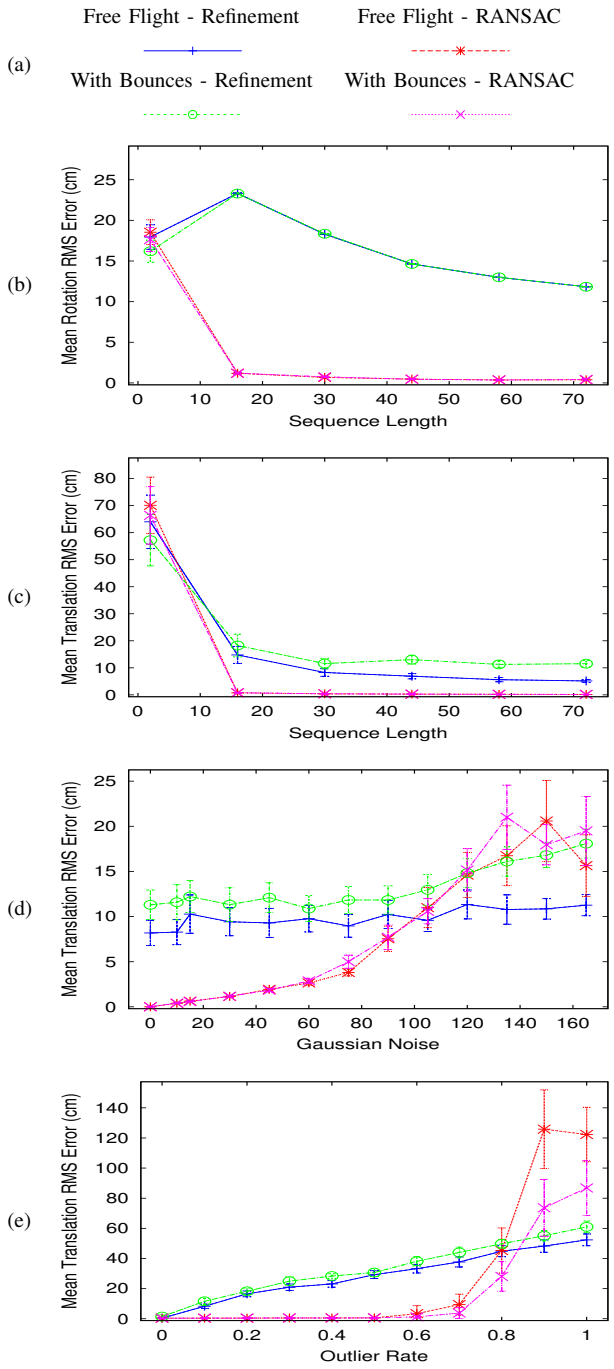

Fig. 2. Results of applying free-flight 3D motion estimation algorithms to automatically generated noisy data. Fifty trajectories are generated and fitted to for each condition. (a) The legend for each of four conditions. A free-flight trajectory is considered as well as a trajectory in which a ground plane induces bounces, and both a basic refinement and a RANSAC algorithm is applied in each case. (b) RMS (root mean square) error in the **rotational** component of the estimated trajectory found by the motion estimation algorithms plotted against increasing **sequence length**. (c) **Translational** RMS error in estimated trajectory found by the motion estimation algorithms, plotted against increasing **sequence length**. In the case of the bouncing object, the number of bounces varies from 1 to 4 with sporadic increments as sequence length increases. (d) **Translational** RMS error with increasing **Gaussian noise**. (e) **Translational** RMS error in the motion estimation algorithms with increasing **outlier rate** (the probability of each timestep being resampled from a uniform distribution). The error bars span a 95% confidence interval. When they're not being varied, the Gaussian noise deviation is $\sigma_{t=}10cm$, the outlier rate $p = 0.1$ and trajectory length 30 time steps. RMS error is averaged over all samples and over the length of the sequence so that these plots illustrate the average error per time step.

then use these image points as observations and generate an observation cost function from them. Observation cost terms and RANSAC inlier thresholds are weighted according to histogram match score for each observation. Each dynamics displacement cost term is given the same weight as an equivalent observation displacement cost term, which is a simple transformation from image units to simulation units (with x_m pixels per metre we get weight x_m^2). Velocity dynamics error terms are weighted according to the heuristic that they can be thought of as differences of virtual displacement random variables, scaled by the frame rate ($1/t_s$), producing weight $(1/2 \cdot 1/t_s)^2$. See table I(e) for calculated values.

We examine the effect on the quality of estimation of each of a handful of different dynamics functions. Standing in for the implicit and explicit dynamics found in the tracking literature, we have a constant displacement dynamics function and a constant velocity dynamics function. We also have a no-dynamics case where the best observation in each frame is taken as-is; in object tracking this would be equivalent to tracking-by-detection.

To demonstrate our experimental cases we have two dynamics models employing the PhysX physics engine (in which we model the floor and as many balls as are in the scene). These cases are the locally parametrised collision dynamics and globally parametrised collision dynamics. See table I(d) for a list of parameters used in the simulation.

We use RANSAC with with Line Search refinement [17] to estimate trajectories. See table I(f-g) for algorithmic parameters. Note that in the case of globally parametrised dynamics any time-point might be required to parametrise a trajectory, since the RANSAC procedure will sample observations at arbitrary time-points. As such, we need to be able to physically simulate backwards in time from such a time-point. We achieve this by inverting object velocities and restitution coefficient and running the simulator forward. However we note that physical parameters such as friction and linear damping are not invertible in the physics simulator. As such, during RANSAC using collision-based dynamics, and while doing gradient descent on the globally parametrised cost function, the estimator can acquire trajectories that it would not normally be capable of simulating. Conversely, as discussed in section II, any locally parametrised dynamics cost only requires a forward dynamics function.

Note that because we use normalised histograms to detect the ball in the video images, the black bag in the video in fig. 3(d) provides a distractor while the ball is occluded. As an artifact of the fact that a ball location is observed at every frame in the image sequence, a mild distractor effect also occurs whenever the ball is not sufficiently visible.

2) Results: Fig. 4 gives the estimation performance of each of the five dynamics models. The constant displacement and constant velocity dynamics fail very badly in many cases because they are unable to select good inliers. As such we also compare the performance of the refinement component of each algorithm by initialising them all with the RANSAC routine employing collision dynamics - see fig. 5.

Fig. 6 contains characteristic examples illustrating the behaviour of constant displacement, constant velocity, and collision dynamics on each video sequence. The two collision-based dynamics perform better on every video except the video where the ball rolls behind an occluder and out the other side again, (c). In this video the constant velocity dynamics performs better. In such a situation the constant velocity dynamics is a sufficient model of object behaviour. The collision-based dynamics fare worse because the ground plane is slightly tilted with respect to the camera while the provided dynamics model assumes that the ground plane is perfectly parallel to the image x-axis.

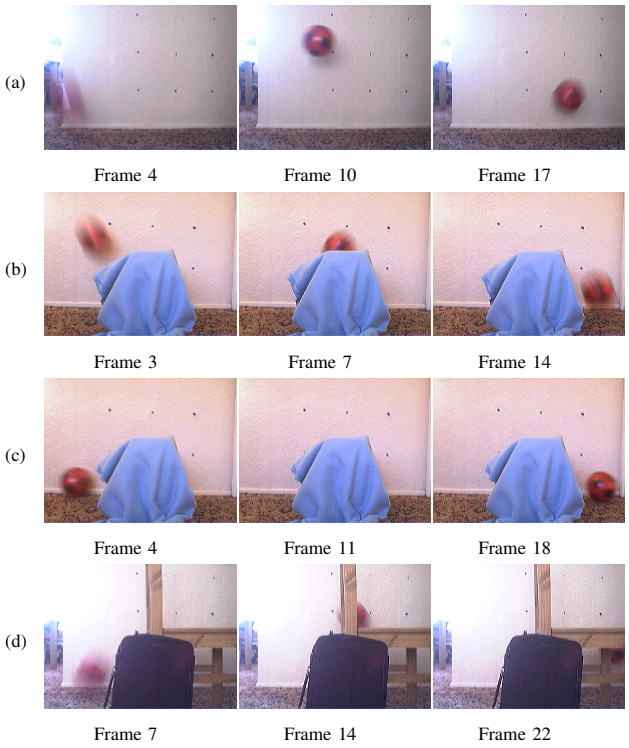

Fig. 3. Image sequences used to test 2D motion estimation. (a) A ball bounces across the field of view with glare, blur. (b) A ball bounces across the field of view, occluded by a stationary object mid-sequence. (c) A ball rolls across the field of view, again occluded by a stationary object. (d) A ball bounces across the field of view, with blur, glare, partial and full occlusion, and in the presence of a strong distractor.

TABLE I

SIMULATOR AND ESTIMATOR PARAMETERS.

3D motion estimation in simple simulation

(a) **Simulation Parameters**			
Ball Radius (cm)	50	Max. Sequence Length (cm)	1000

(b) **Refinement parameters**		(c) **RANSAC parameters**	
Maximum Iterations	100	Maximum Iterations	100
Finite Differences Step	10^{-3}	Inlier Threshold (cm)	140
Stopping Tolerance	10^{-7}	Minimum Inlier Count	$(t_f - t_i)/5t_s$

Motion estimation from colour histogram matches

(d) **Simulation Paramaters**			
Coeff. Restitution	0.8	Linear Damping	0.2
Coeff. Static Friction	2.0	Ball Mass (measured) (g)	110.0
Coeff. Dynamic Friction	0.1	Radius (measured) (cm)	10.5

(e) **Cost Paramaters**			
Dyn./Obs. Cost (cm^{-2})	$1.2 \cdot 10^5$	Vel./Disp. Cost (s^{-2})	225

(f) **Refinement paramaters**		(g) **RANSAC paramaters**	
Maximum Iterations	400	Maximum Iterations	400
Finite Differences Step	10^{-5}	Inlier Threshold (pixel dist.)	40
Stopping Tolerance	$3 \cdot 10^{-6}$	Minimum Inlier Count	15

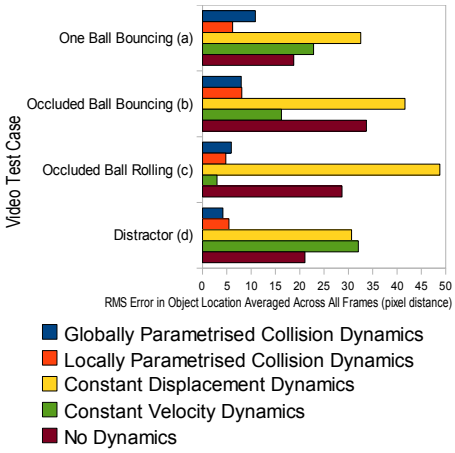

Fig. 4. Initialisation and refinement performance of 2D motion estimation from colour histogram algorithms: Performance measured in RMS error in pixel distance from labelled object location, averaged over all frames of each of five algorithms on each of four image sequences. Shorter bars indicate better performance and lower error. In these results, each dynamics model is used both in the initialisation and refinement phases. (a-d) refer to the same image sequences as shown in fig. 3.

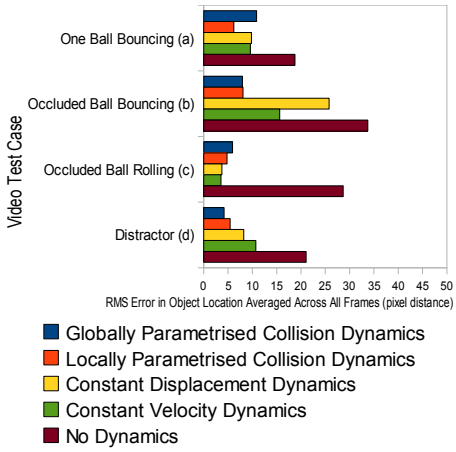

Fig. 5. Refinement only performance of 2D motion estimation from colour histogram algorithms: Performance measured in RMS error in pixel distance from labelled object location, averaged over all frames of each of five algorithms on each of four image sequences. Shorter bars indicate better performance and lower error. In these results, the collision-based dynamics is used to select an initial set of inliers and to initialise the refinement procedure which then uses each of the dynamics models listed in the legend. (a-d) refer to the same image sequences as shown in fig. 3.

Note also that the locally parametrised collision dynamics does better than the globally parametrised collision dynamics in those videos where observation data is useful in correcting the subtle mismatches between the dynamics model and the behaviour of objects in the real-world (a & c). Where there are lots of distractors and occlusion (d), however, the global collision dynamics is able to compensate more for the bad data than local collision dynamics can.

In video (a) the constant displacement dynamics makes a characteristic error of estimating the ball position near to observed positions nearby in time, and the constant velocity dynamics estimates the ball to be travelling through the floor. The collision dynamics is able to compensate somewhat by giving credence to the hypothesis that the ball may have bounced. Video (b) shows both non-collision dynamics unable to benefit from hypotheses involving the bouncing of the ball behind the obstacle. Videos (c) and (d.i) show the constant velocity dynamics succeeding but constant displacement dynamics unable to initialise a trajectory that moves a long distance between observations. Video (d.ii) shows constant velocity dynamics unable to initialise a trajectory that involves the ball changing direction while occluded.

Informal experiments suggest that the performance of the algorithm is much more sensitive to the RANSAC inlier threshold parameter than to tunable parameters in the physics model such as the coefficient of restitution, but only with these relatively simple scenarios.

3) Discussion: The data confirm our hypothesis that a more sophisticated dynamics model, particularly involving collisions, is able to substantially improve motion estimation, particularly when observation data is absent or misleading, by quickly finding good initial estimates, and by guiding the search for trajectories away from distractors and towards feasible trajectories. However, this improvement is contingent on a good match between model and world.

In these experiments, rough manual estimates of simulation parameters such as restitution coefficients were sufficient for good results. However preliminary experiments have shown us that, to be useful generally, this approach requires increasingly fine specification of these parameters when dynamics are more sensitive, such as with ball-ball collisions.

V. Summary and Future Work

A. Summary

We present a motion estimation framework that allows the incorporation of arbitrary dynamics models. We demonstrate that the use of a dynamics as provided by an off-the-shelf physics simulator is able to improve the accuracy of motion estimation when the model and the world match well.

B. Future Work

There are open questions as to which simulation parameters the estimation is sensitive to, which simplifications may be made with little impact on performance, which model aspects are already impacting accuracy by deviating from real-world behaviour, and to what extent chaotic dynamics can be handled by increasing the accuracy of simulation parameters. We would like to test the proposal that allowing refinement of parameters of the physics engine, or local surface shape parameters, will lead to a more accurate estimation procedure. It might be possible to use the provided physics model to bootstrap the learning of a more generic dynamics model or one more like those used by humans [18].

We are currently working on image sequences involving two interacting identical balls, again in 2D, much like in the work of Chang, et al. 2005 [19]. This work involves the necessary addition of layers in the depth direction and a data association framework. We predict that collision dynamics will improve performance in a number of scenarios, for instance, when an object collides with another and loses

energy as a consequence, thereby disambiguating the nature of the interaction.

There is also room for incorporating a more intelligent way of handling observations, such as using the colour histogram match score directly in the observation function; better observations may obviate somewhat the effect of intelligent dynamics.

Of course, it would be useful to extend this work to 3D and to arbitrary objects. Our next effort will be in the area of robotic manipulation where we will adapt a recursive filter such as a particle filter or an unscented Kalman filter to use the physics engine in a probabilistic forward dynamics model. Particular concerns are the non-linearity of the physics model and potentially high number of dimensions when considering velocity state parameters and multiple objects.

VI. ACKNOWLEDGEMENTS

We gratefully acknowledge the contribution of the EU FP6 IST Cognitive Systems Integrated Project (CoSy), FP6-004250-IP, the EU FP7 IST Project CogX FP7-IST215181, University of Birmingham School of Computer Science and the UK Overseas Research Students Awards Scheme. Many thanks to Zeyn Saigol for his careful reading.

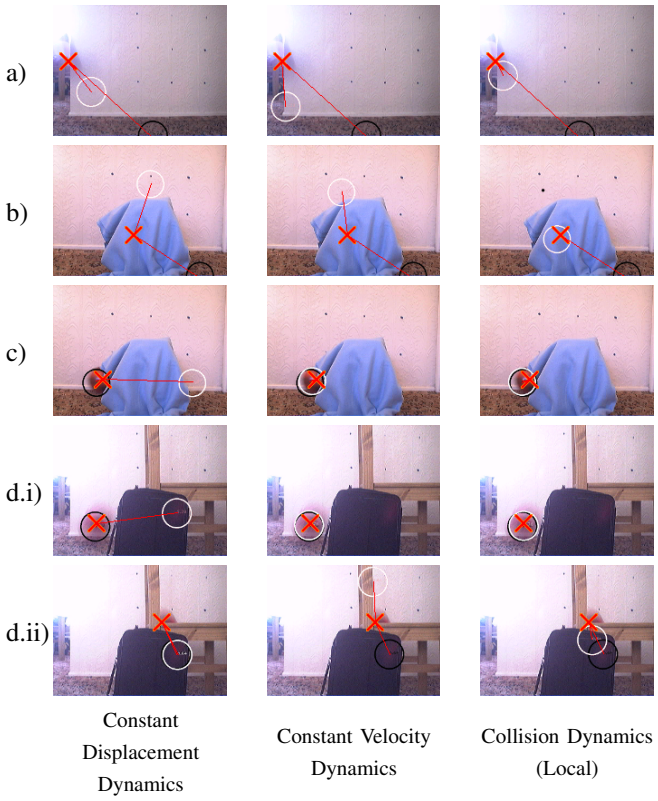

Fig. 6. Some examples of success and failure of 2D motion estimation from colour histogram algorithms. Three dynamics models are illustrated: constant displacement, constant velocity, and collision dynamics based on local parametrisation. Each row shows one time-point during a test image sequence, and each column corresponds to a dynamics model as applied to that image sequence. The white circle represents the estimated ball location as determined using the dynamics model; the black circle represents the estimated ball location as determined in the absence of any dynamics model; the red cross represents the pre-labelled (by hand) ball location, connected by red lines to the estimated ball locations. The first two rows are from examples pre-initialised using collision dynamics while the remaining three rows use the same dynamics model for initialisation as for refinement. (a) One bouncing ball, frame 3. (b) Ball bouncing while occluded, frame 5. (c) Ball rolling while occluded, frame 6 (here it can be seen that constant velocity dynamics performs best). (d.i) Ball bouncing behind distractor, frame 7 (in this frame constant velocity dynamics performs almost as well as the collision-based dynamics but constant displacement dynamics has failed because of the distractor). (d.ii) This is the same case at a later frame: ball bouncing behind distractor, frame 15 (by this point in the sequence the constant velocity dynamics has also failed).

REFERENCES

[1] D. Kragic, A. T. Miller, and P. K. Allen, "Real-time tracking meets online grasp planning," in *Proceedings IEEE International Conference on Robotics and Automation*, Seoul, Republic of Korea, 2001, pp. 2460–2465.

[2] C. Wren and A. Pentland, "Dynamic models of human motion," in *Proceedings IEEE International Conference on Automatic Face and Gesture Recognition*, 1998, pp. 22–27.

[3] C. Sminchisescu and B. Triggs, "Estimating articulated human motion with covariance scaled sampling," *International Journal of Robotics Research*, vol. 22, no. 6, pp. 371–393, 2003.

[4] M. Vondrak, L. Sigal, and O. C. Jenkins, "Physical simulation for probabilistic motion tracking," in *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Los Alamitos, CA, USA: IEEE Computer Society, 2008, pp. 1–8.

[5] J. Popovic, S. M. Seitz, M. Erdmann, Z. Popovic, and A. Witkin, "Interactive manipulation of rigid body simulations," in *SIGGRAPH*, 2000, pp. 209–218.

[6] K. S. Bhat, S. M. Seitz, J. Popovic, and P. K. Khosla, "Computing the physical parameters of Rigid-Body motion from video," in *Proceedings of the European Conference on Computer Vision: Lecture Notes in Computer Science*, vol. 2350, 2002, pp. 551–565.

[7] G. Young, R. Chellappa, and T. Wu, "Monocular motion estimation using a long sequence of noisy images," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 1991, pp. 2437–2440.

[8] T. J. Broida and R. Chellappa, "Estimating the kinematics and structure of a rigid object from a sequence of monocular images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 6, pp. 497–513, 1991.

[9] X. Hu and N. Ahuja, "Long image sequence motion analysis using polynomial motion models," in *IAPR Workshop on Machine Vision Applications*, 1992, pp. 109–114.

[10] M. Chan, D. Metaxas, and S. Dickinson, "Physics-Based tracking of 3D objects in 2D image sequences," in *Proceedings International Conference on Pattern Recognition*, 1994, pp. 432–436.

[11] D. Metaxas and D. Terzopoulos, "Shape and nonrigid motion estimation through physics-based synthesis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, no. 6, pp. 580–591, 1993.

[12] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," *Intel Technology Journal*, vol. Q2, 1998.

[13] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Eur. Conf. on Computer Vision, ECCV'2002, LNCS 2350*, Copenhaguen, Denmark, June 2002, pp. 661–675.

[14] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 5, pp. 564–577, 2003.

[15] A. Naeem, S. Mills, and T. Pridmore, "Structured combination of particle filter and kernel Mean-Shift tracking," in *International Conference Image and Vision Computing New Zealand*, vol. 21, 2006.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381–395, 1981.

[17] R. Fletcher, *Practical Methods of Optimization - Volume 1*. Chichester: John Wiley & Sons, 1980.

[18] R. Grzeszczuk, D. Terzopoulos, and G. Hinton, "NeuroAnimator: fast neural network emulation and control of Physics-Based models," in *Computer Graphics Proceedings, Annual Conference Series*. Orlando, Florida: ACM SIGGRAPH, 1998, pp. 9–20.

[19] C. Chang, R. Ansari, and A. Khokhar, "Multiple object tracking with kernel particle filter," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2005.