

USING PHYSICS ENGINES TO TRACK OBJECTS IN IMAGES

F. de Chaumont^{a}, A. Dufour^a, P. Serreau^b, J. Chabout^b,
S. Münter^d, F. Frischknecht^d, S. Granon^{bc} and J.-C. Olivo-Marin^a*

Institut Pasteur, 25-28 rue du Dr. Roux, 75015 Paris, France

^aUnité d'Analyse d'Images Quantitative, CNRS URA 2582

^bUnité de Neurobiologie Intégrative des Systèmes Cholinergiques, CNRS URA 2182

^cNeurobiologie de l'Apprentissage de la Mémoire et de la Communication, CNRS 8620, Orsay, France

^dDepartment of Parasitology, Hygiene Institute, University of Heidelberg MS, Heidelberg, Germany

ABSTRACT

This paper tackles the problem of tracking multiple articulated objects undergoing frequent contacts in a video sequence. Conventional tracking methods usually fail to distinguish objects during contact, implying the use of disambiguation techniques to recover the identity of each object. Moreover, such methods do not provide detailed shape information at the articulation level. We address these limitations by proposing a novel approach to track multiple articulated objects using the mean-shift technique and physics engines. By defining a model of each object using geometrical primitives and physical constraints, we exploit the physics engine force solver as a control layer of the mean-shift process and follow the model and its deformation along the sequence. The method is applied to track mice observed with a webcam and two sporozoites observed in reflection interference contrast microscopy, highlighting the flexibility and genericity of the framework.

Index Terms— Tracking, Mean-shift, Physics engines

1. BACKGROUND

1.1. Multi-object tracking

Multi-object tracking in video sequences is a field of active research in computer vision, with numerous applications to scene surveillance [1, 2, 3, 4]. Throughout the literature, mean-shift based techniques [3, 5] are widely used to track whole objects in real-time in natural environments thanks to their computational efficiency. In biological imaging, such methods have been employed to track multiple deforming cells throughout a sequence [6]. Yet, mean-shift approaches have two important limitations. Firstly, tracking ambiguities quickly arise when similar objects come into contact, where concurrent models are attracted toward a common local maximum and fuse until the objects separate again. This association problem is common in many other applications, and additional methods must be employed to solve these ambiguities (e.g., [7]). Secondly, mean-shift and CAMShift [8] approaches are able to track objects as a whole, accommodating global object transformations such as scale, orientation or pose changes, but do not quantify local shape information

and thus cannot monitor deformations at the articulation level. Despite their efficiency for multi-object tracking, this remark also holds for active contour methods [9], which are also computationally more demanding.

In this paper, we address both limitations by proposing a novel approach for multi-object tracking using physics-constrained mean-shift. The principle is to define a model of the target using geometrical primitives that are linked via physical constraints. Each primitive of the model is then linked to the image data using a standard mean-shift process, and the global model deformation is performed in real-time using a physics engine solver. As a result, the proposed approach is able to: a) track objects in a very robust way, b) maintain the identity of multiple touching objects, c) provide local shape information at the articulation level through each of its geometrical primitives. The remainder of this paper is organized as follows: we first give below a short overview of physics engine, and then describe the proposed method for tracking articulated objects in section 2. Section 3 discusses the performance of the method on two sample applications. Finally, section 4 concludes the paper by discussing the genericity of the proposed approach and its potential applications to other types of objects.

1.2. Object tracking with physics engines

1.2.1. Rise of the physics engines

Physics engines are software or hardware toolkits dedicated to the realistic simulation of physical systems governed by Lagrangian dynamics. They first appeared over 30 years ago, but only received extensive attention in the last decade. Due to technical limitations, early engines were tailored for very specific applications, and had therefore very limited interest. Nowadays, their evolution is mostly driven by the computer-gaming industry, striving to produce ever more physically realistic video games. As a result, numerous commercial and free APIs have been developed in recent years for the simulation of highly complex systems, addressing delicate problems such as collision detection between articulated objects in an efficient way. Many engines are now integrated within modern graphics cards, benefiting from multiple hardware optimizations and parallel implementations which are crucial for real-time applications. The goal of this paper is to show how such engines can be used to perform object tracking in video sequences. By representing the objects using physically constrained models, and by linking these models to

*This work is funded by CNRS and Institut Pasteur. E-mail: {chaumont,adufour,jcolivo}@pasteur.fr. We acknowledge Nicolas Chenouard and Marcio de Moraes Marim (Institut Pasteur) for valuable discussions.

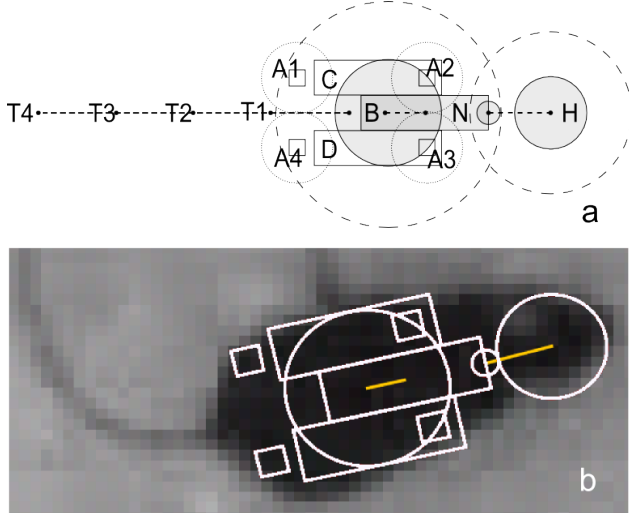


Fig. 1. Physics model of a mouse. (a): description of the model bodies (solid lines) and the boundary of their attractive maps (dashed lines). Grey bodies are not allowed to overlap with grey bodies of another model. (b): final model matching a mouse in a sequence frame.

the image data, the engine solver is able to track the evolution of all objects simultaneously, even if they come into contact over time.

1.2.2. Defining a physics model

Physics engines define a model \mathcal{M} in a world \mathcal{W} . A model $\mathcal{M}_{O,C}$ (Fig. 1a) is defined by a set $O = \{o_i\}_{i=1..n}$ of basic primitive objects (also called *bodies*), characterized by a mass, a damping (i.e. friction) factor, and an anchor where displacement forces will be applied. Basic primitives include rectangles, circles and other convex polygons, while more complex objects can then be constructed as combinations of these primitive bodies. Bodies are linked together by constraints called *joints*. These joints are at least of four types: distance, angle, elastic and slider. Collision between individual bodies and between independent groups of joint bodies are both handled by the physics engine automatically, using various built-in algorithms that either favor computation time or physical realism.

1.2.3. Handling model movements

There are traditionally two types of physics engines, depending on how the bodies are handled. *Rigid-body*-engines consider bodies through their centers only, yielding simple management, but requiring an additional rotation parameter to account for all types of movements. *Mass-aggregate*-engines handle bodies as a set of little masses (i.e. a box is described by 8 masses representing its corners). Using this latter representation, all movements can be described without any additional parameter, though more computations are required to induce a correct displacement of all components of the body. We shall see later that the mass-aggregate approach allows us to perform motion prediction in a somewhat straightforward manner (see section 2.3.2).

In the following section, we describe how the initial tracking problem can be expressed as a set of local mean-shift processes applied to individual bodies, and combined into a global physically-constrained system to exploit the physics engine solver. To the best

of our knowledge, this is the first time that such an approach is taken, thereby bridging the gap between physics engines and image-based object tracking.

2. PHYSICS-CONSTRAINED MEAN-SHIFT

2.1. An image-based physics model

In order to exploit image data to attract the physics model toward the object position, we define a relation between each body anchor and the image data. An anchor a is thus defined by a point p_a and an attraction map M_a computed from the original image I . Depending on the image information that is exploited, one can define multiple attraction maps, and use a different map for each anchor. Each map is then restricted to a sub-region of the image centered on the anchor. Finally, for each anchor a , one computes the resulting attraction force $\vec{f}(a)$ as follows:

$$\vec{f}(a) = \sum_{q \in M_a} M_a(q) \overrightarrow{p_a q}, \quad (1)$$

where q describes all pixels within attractive map.

The final deformation step involves computing \vec{f} for all objects in \mathcal{W} , and feeding these forces to the physics engine solver, which will take care of combining these forces with the physical properties of each object as well as additional physical constraints. The deformation is an iterative process, therefore attraction forces must be recomputed at every iteration. Convergence can then be detected automatically using various stopping criteria. A classical strategy is to detect when the system reaches a steady-state, where the speed of all models falls below a small value ϵ . Note that if each model can be converted to a probabilistic distribution, another way to detect convergence is to measure the distance between each model distribution and some reference distribution, e.g. using the commonly-used Bhattacharyya distance [3].

2.2. Relation with the mean-shift approach

The principle of mean-shift tracking [3] is to move a model toward a local maximum defined by a kernel. In our method, the model corresponds to a body of our physics model, and the associated kernel is a gate analogous to the attraction map associated to the body anchor. The iterative process that seeks for the local maximum is similar. However, our approach incorporates physical constraints between bodies of a same physics model, thereby improving object detection and tracking using a higher level of semantic information.

2.3. Application to multi-object tracking

The appealing interest of the physics model approach is that an arbitrary number of models can be defined in the same world and deformed simultaneously using the physics engine solver. One can thus perform concurrent multi-object tracking, where each model follows a given target in the image according to the displacement of each of its constitutive objects. In case two objects come into contact, ambiguities related to model overlapping are automatically detected and handled by the physics engine through efficient collision detection and feedback strategies.

2.3.1. Dealing with object proximity

While model overlapping is forbidden, it may occur that anchors of same nature are sufficiently close such that their attraction maps do

overlap. Consequently, an identical portion of image information is exploited to compute the movement of concurrent anchors, potentially attracting the models toward wrong positions (as illustrated in Fig. 2). We remove the ambiguity by masking all attractive maps of same nature such that all pixels shared by two or more maps are not involved in the computation of \vec{f} . For any given anchor a_i we compute the intersection between its attractive map M_{a_i} and the attractive map of all other anchors a_j (provided the maps are of same nature), and exclude this intersection from M_{a_i} . Eq. 1 thus becomes:

$$\vec{f}(a) = \sum_{q \in M_a^*} M_a^*(q) \vec{p}_a q, \quad \text{with} \quad (2)$$

$$M_{a_i}^* = M_{a_i} / \bigcup_{a_j \neq a_i} M_{a_j}$$

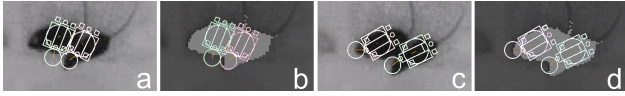


Fig. 2. Matching mice in close contact without (a,b) and with proximity handling (c,d). With proximity handling, the overlap between attractive maps of same nature is removed from the computations, yielding a correct ajustement of the models.

2.3.2. Motion Prediction

For simple applications where the object does not move significantly from a frame to the next, tracking is performed straightforwardly by using the resulting model on a given frame to initialize its position in the next frame. Nonetheless, if the acquisition frame rate is not sufficient, the object may move fast enough such that its new position is no longer reachable by the model, as it is no longer visible within the attractive maps.

To cope with this situation, and more generally to reduce convergence time, we take advantage of mass-aggregate engines to predict the position of the model for each frame but the first. We recall that in such engines the whole object undergoes complex movements as a result of the constrained linear motion of each component. We exploit this motion information by computing the speed vector for each anchor from frame $t - 1$ to frame t , and extrapolate this speed to obtain a predicted position on frame $t + 1$.

3. RESULTS

3.1. Sporozoites tracking

This experiment consists in tracking *plasmodium berghei* sporozoites observed in reflection interference contrast microscopy¹. Although the sporozoites do not exhibit major shape changes, the difficulty with such images is that local structures may not be visible, causing local detection methods to fail (c.f. Fig. 3). In the present case, we rely on the prior knowledge that this type of sporozoite moves in a pseudo-circular fashion. To improve the tracking performance, we incorporate this shape information directly in the physics model, thereby denoting the flexibility of the framework.

¹See <http://www.bioimageanalysis.org/9248> for the full video with tracking results

The sporozoite model thus comprises a set of anchors linked to each other by a distance and an angular joint, imposing a constant curvature to the model during its displacements. The sporozoites are then matched by computing attractive maps for each anchor from a wavelet transform of the image [10]. As a result, the global model successfully tracks the whole sporozoites, even if some parts are invisible. Fig. 4 plots the measured sporozoites displacements in pixels for each sequence frame.

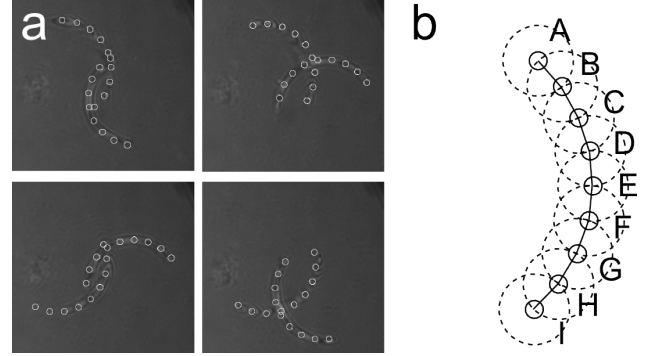


Fig. 3. Tracking of two sporozoites observed in reflection interference contrast microscopy (200 frames). (a): sample frames of the original sequence. (b): physics model of the sporozoite.

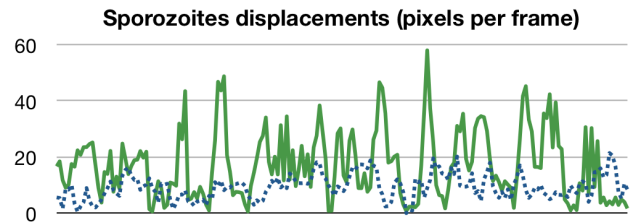


Fig. 4. Tracking results on the sporozoite sequence. Each line represents one sporozoite. Displacement values are given in pixels for each frame.

3.2. Multiple mice tracking

Our goal is now to extract behavioral statistics from the co-habitation of two mice evolving in a rectangular cage observed with a webcam. More particularly, we focus on labeling three types of events: oral-oral contact and oral-genital contacts (in both directions). To do so, we must be able to determine at all times the position and orientation of each mouse. This work was previously done manually with a timer, and an important number of playbacks were necessary to label the different events. Moreover, these events having a very short duration, they are often labeled in over-sized time frames. We describe the mouse physics model below and then present quantitative results on a sample sequence¹.

3.2.1. Physics model of the mouse

The physics model of our mouse (illustrated in Fig. 1) is defined as follows: the bodies H, B and N are the head, the belly and the neck of the mouse, respectively. As the mouse can elongate or contract itself, we allow the neck to slide along the belly, constrained between parts C and D. A sliding joint is added to bound the distance

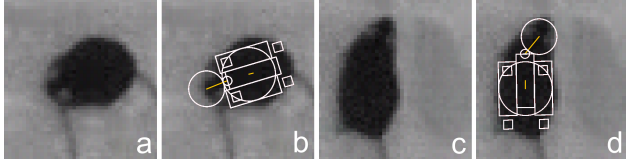


Fig. 5. Mouse with a contracted posture (a) and with the head on the side (c). (b,d): final mouse model super-imposed on the original images.

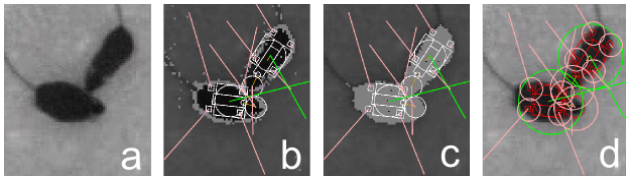


Fig. 6. Force vectors computed for each model anchor. (a): original image. (b,c): force vectors (c.f. Eq. 2) super-imposed on a gradient map (b, green vectors) and binary map (c, pink vectors) of the image. (d): vectors and boundaries of their attractive maps.

between the belly and the neck. The head is a simple circle and is linked to the neck by a distance joint. In order to prevent overlap between different mice models, we impose that all bodies colored in gray may not overlap with similar bodies of another model (see Fig. 6 for an illustration of mice contact situation). Anchors H and A1 through A4 exploit an attractive map based on intensity gradients, while B exploits an attractive map that is a binary image highlighting dark image regions (i.e. reflecting mice). Dashed circles depict the boundary of the attractive maps. Finally, T1 to T4 are bodies defining the mouse tail, and are linked by a distance joint. In the current model, they are not linked to any attractive map. The final model is shown super-imposed on a mouse image in Fig. 6.

3.2.2. Results

Tracking results are illustrated in Figs. 2, 5 and 6. The proposed method is able to automatically detect the position and orientation of the mice even when they are in close contact, and compute the proximity of the head and posterior of each mouse to label the events in an efficient way. We have applied the method on a 530 frame sequence, acquired at a rate of 12.5 fps (frames per second), i.e. about 42 seconds long. The final timeline of events for this sequence is presented in Fig. 7. Thanks to its computational efficiency, the method is able to process the whole sequence in 55 seconds, i.e. yielding a processing rate around 10 fps², which is very close to real-time in the present case.



Fig. 7. Timeline of the labeled events found on a 40-seconds sequence (530 frames). In this particular sequence, one can note that the 'head 1 - genital 2' event never occurs.

²Computation using the Java Phys2D API on a 2.5Ghz Opteron processor without GPU acceleration.

4. CONCLUSION

In this paper we presented a novel approach to track multiple articulated objects in 2D video sequences using geometrically constrained models and physics engines. The method describes the tracking problem a set of multiple local mean-shift processes applied to individual bodies, which are then linked together into a physics model. The physics engine solver is used to solve all these local processes while handling user-defined constraints such as collision between different models in a fast and fully automated fashion. Our future work will focus on improving the computation time using hardware acceleration, in order to reach real-time computation and allow videos to be processed and labeled directly during their acquisition.

5. REFERENCES

- [1] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research: Part C*, vol. 6, pp. 271–288, 1998.
- [2] I. Cohen and G. Medioni, "Detecting and tracking moving objects for video surveillance," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1999.
- [3] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, may 2003.
- [4] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1208–1221, September 2004.
- [5] C. Yang, R. Duraiswami, and L. Davis, "Efficient mean-shift tracking via a new similarity measure," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- [6] O. Debeir, P. Van Ham, R. Kiss, and C. Decaestecker, "Tracking of migrating cells under phase-contrast video microscopy with combined mean-shift processes," *Medical Imaging, IEEE Transactions on*, vol. 24, no. 6, pp. 697–711, 2005.
- [7] A. Genovesio, T. Liedl, V. Emiliani, W.J. Parak, M. Coppey-Moisand, and J.-C. Olivo-Marin, "Multiple particle tracking in 3D+T microscopy: methods and application to the tracking of endocytosed quantum dots," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1062–1070, 2006.
- [8] G.R. Bradski, "Real time face and object tracking as a component of a perceptual user interface," *IEEE Workshop on Applications of Computer Vision*, pp. 214–219, 1998.
- [9] C. Zimmer and J.-C. Olivo-Marin, "Coupled parametric active contours," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 27, no. 11, pp. 1838–1842, 2005.
- [10] J.-C. Olivo-Marin, "Extraction of spots in biological images using multiscale products," *Pattern Recognition*, vol. 35, no. 9, pp. 1989–1996, 2002.