Tracking T and B cells from two-photon microscopy imaging using constrained SMC clusters

D. Olivieri1*, J. Faro2,3, I. Gomez-Conde1, C. E. Tadokoro3

1University of Vigo, ESEI, 32004 Ourense, Spain, http://www.uvigo.es

2University of Vigo, Department of Biochemistry, Genetics and Immunology, 36310 Vigo, Spain, http://www.uvigo.es

3Instituto Gulbenkian de Ciencia, 2781-901 Oeiras, Portugal, http://www.igc.gulbenkian.pt

Summary

This paper describes a novel software algorithm, called constrained Sequential Monte Carlo (SMC) clusters, for tracking a large collection of individual cells from intra-vital two-photon microscopy image sequences. We show how our method and software tool, implemented in python, is useful for quantifying the motility of T and B lymphocytes involved in an immune response vs lymphocytes under non immune conditions. We describe the theory behind our algorithm and briefly discuss the architecture of our software. Finally, we demonstrate both the functionality and utility of software by applying it to two practical examples from videos displaying lymphocyte motility in B cell zones (follicles) and T cell zones of lymph nodes.

1 Introduction

Recent advances in intra-vital multi-photon laser microscopy have made it possible to visualize cellular motion within lymphoid tissues in real time and in living animals [1, 2]. In order to extract quantitative information from such image sequences, which are helpful for understanding immune processes at cell and population levels, new software tools are needed that can accurately track the position of individual cells over time and in 3-dimensions for a large number of cells. Due to the complexity of this problem, existing software tools are prone to tracking errors, and often biologists must resort to manually tracking individual cells, thereby putting a practical limit on the statistics that can be obtained. Because of this, studies tend to rely only upon a small number of cell tracks and thus there is a large debate about the validity of certain biological interpretations. Therefore, a software analysis package would be an important contribution to the growing set of immunoinformatics tools [3]. In this paper, we describe a new set of algorithms based upon constrained particle filters as well as our software implementation for tracking a large collection of individual cells in a complex cellular environment obtained from two-photon microscopy. In a more general setting, our tool could be useful for biologists who require detailed quantitative motility analysis of many individual cells.

*To whom correspondence should be addressed. Email: dnolivieri@gmail.com
As a motivation of this work, there are several important questions related to the migratory behavior of $B$ and $T$ lymphocytes populations, which are responsible for specific and acquired immunity. In particular, a central question concerns the observed differences of cell migration patterns and formation of dynamical structures within secondary lymphoid tissues (e.g. spleen and lymph nodes) during homeostatic conditions as compared with those observed during immune responses. In these dynamical structures, $B$ and $T$ cells become spatially segregated in distinct, but adjacent zones, by virtue of their different sensitivities to sets of chemokine gradients. Moreover, as compared to homeostatic conditions (i.e., in the absence of an immune response), the migratory patterns of dendritic cells (DC), antigen-specific $T$ helper ($T_h$) cells, and $B$ cells change dramatically during immune responses to protein antigens, in a manner that greatly enhances the DC-Th cell and cognate $T_h$-$B$ cell interactions. The former encounters are facilitated in the center of $T$-cell zone [4, 5, 6, 7, 8] while the latter interactions occur on the T-cell/B-cell zone border [9, 10, 11].

Subsequently, a fraction of activated, antigen-specific $B$ and $T$ cells migrate to the adjacent $B$-cell zone or follicle, initiating the formation of germinal centers (GC). Germinal centers are temporary structures with an apparent basal-apical axis where activated $B$ cells proliferate intensely and interact with antigen retained on follicular DC cells (FDCs) and with follicular $T_h$ cells. This later type of interaction induces within $B$ cells a hypermutation process thereby generating many mutant antibodies. A selection mechanism within GCs favors those $B$ cells with antibody mutations that enhance the antibody affinity for antigen, leading to an increase of affinity, or affinity maturation.

Recent research by various groups has suggested that an underlying dynamic mechanism responsible for the spatial zoning of GCs must play an important role in the affinity maturation process. This mechanism is thought to be linked to the changing migratory patterns of $B$ and follicular $T_h$ cells during the relatively short time life of GCs. For example, one model known as the cyclic re-entry model posits that, within mature GCs, affinity maturation requires frequent trafficking of $B$ cells between the dark and light zones [12, 13]. Other studies, hypothesize that motility of $B$ cells is directed instead of following a random walk [14, 15]. Yet other studies have suggested that there is a competition for $T$ cells that play a key role in the affinity maturation and selection process [16, 17]. Studying lymphocyte migratory behavior in GCs should provide direct tests for those currently competing models of the selection process. Thus, in order to firmly establish a link between $T$ and $B$ cell migratory patterns, lymphocyte activation status, and particular immune mechanisms, accurate cell tracking of large cell samples is required.

Yet, present methods are hampered by the fact that real experimental imaging data from two-photon microscopy contain many complicated artefacts and cells are constantly crossing and merging. Thus, in order to automatically analyze migration data for $B$ and $T$ cells for in-vivo experimental conditions, a software tool must be robust, simultaneously track a large collection of cells at multiple focal depths (the so-called $z-$stack) over time (video sequences), while capable of removing spurious motion inherent within in-vivo imaging. In this paper, we describe the present status of our algorithm and software tool for treating this problem, which are based upon constrained clusters of particle filters. Our approach differs significantly from other software and algorithms applied to this problem domain. Thus, we lay the foundations of the method and show results using experimental datasets.
2 SMC Based Tracking

Tracking objects has a long history in the field of computer vision. In general, foreground objects must be distinguished from complex backgrounds. Indeed, one of the most widely cited algorithms in computer science in recent times is the so-called Snake (or active contour) algorithm first introduced by Kass, Witkin, and Terzopoulos [18] for segmenting and tracking objects by minimizing the internal and external energy of a contour with respect to properties of the underlying image pixels. This algorithm, has been improved over the years and has been combined with other powerful ideas such as gradient vector flows, diffusion, and more general level set methods. An inconvenience for fully automated algorithms is that given a static image, these segmentation algorithms require interactive object selection criteria. Image sequences (videos) provide a significant advantage in that they allow moving objects to be discerned from the background by frame differencing or the optical flow field of the object. Despite the attractive appeal of these methods, complications arise in many practical situations from noisy and moving backgrounds.

In order to provide more robust tracking and obviate crossing ambiguities, information about the state space dynamics of moving objects could be incorporated in tracking algorithms. Algorithms of this type are based upon Kalman filters [19], and a stochastic version, referred to as Sequential Monte Carlo, which is a fully probabilistic dynamic model that obtains the most likely position of the object as a function of time by randomly sampling the image with some criteria such as color, shape, or contour. The Sequential Monte Carlo (SMC) method, or so-called particle filter algorithm, was first described by Gordon in 1993 [20] and later implemented for computer vision by Isard and Blake [21]. Recently, much attention has been given to the SMC, with development of specific kernels and use with the Markov Chain Monte Carlo (MCMC) algorithm.

For the particular problem of tracking living cells, several commercial and open source software solutions are available. A comparison of existing software for tracking cells from phase-contrast microscopy has been described by Hand et al. [22], where classic methods based upon frame differencing as well as image registration (or alignment) are treated. Amongst the most widely known open-source software tools for automatic tracking of cells are CellTrack, and the plugin for ImageJ, called MTrack2, while well known commercial tools are Imaris, Prisma graphics, and Velocity. While these tools have been developed for high contrast microscopy (using fluorescently labelled or phase contrast cells), the image registration methods used are not reliable for lower contrast imaging found in in-vivo two photon images sequences. Ambiguities arise when cells cross or join along a trajectory since the cell membrane is not well differentiated. In these cases, practitioners normally resort to manual based tracking of individual cells, since present commercial and open source software solutions do not provide sufficient reliability.

As an alternative to cell tracking software presently available, we propose in this paper a novel method based upon the SMC algorithm that is specifically constructed for treating the cases of cell sticking/joining and crossing. Our method consists of tracking with a set of weakly interacting clusters of constrained SMC nodes that move together with cells by sampling the underlying image, while approximately conserving both boundary and area constraints through probabilistic weighting. Specifically, the method derives future trajectories based upon a maximum likelihood sampling within the neighborhood of predicted trajectories of the object to be tracked.
One potential advantage of our method is for the problem of cell merging, which is a common event for cell tracking. In this case, clusters of particles maintain their area and are endowed with a non-overlap condition through cluster-cluster repulsive force mimicking the non-overlap condition of the cell membrane. Intuitively, the constrained SMC clusters act as soft balls slightly bouncing off one another when a cell merge event occurs, thereby providing a natural ballooning effect and maintaining areas of each cell. Another potential advantage of this method as compared with image registration methods is for three-dimensional tracking of cells, since individual particle filters can scale with the size of the object to be tracked over the trajectory.

### 2.1 Definition of the SMC method

Figure 1 illustrates the use of particle filters for tracking cells. SMC algorithms provide a general framework for a large number of practical development choices that must be made for particular problem domains. In general, dynamical system can be modeled as Hidden Markov chains [23], with the unknown states written as $x_{0:t} = \{x_0, \cdots, x_t\}$ and observations given as $y_{1:t} = \{y_1, \cdots, y_t\}$, where the subscripts represent discrete time. For the application of tracking, the observations represent information derived from the underlying pixels, while the hidden states are the object’s position and velocity. With these definitions, the dynamics of a system can be completely described by: (a) the initial distribution $p(x_0)$, (b) an evolution model $p(x_t|x_{0:t-1}; y_{1:t-1}) \approx p(x_t|x_{t-1})$ (which is the first-order Markov assumption), and (c) a likelihood function $p(x_t|y_{1:t-1})$. 

---

**Figure 1:** Concepts for particle filter tracking of cells: (a) representative particle samples of individual regions of interest (ROI) of the image frame at time $t$, (b) diagram of particle filter ROI samples along the trajectory from $x_t$ to $x_{t+1}$, and illustrating histograms obtained for each particle ROI. Observations are compared against a reference histogram, obtained upon particle node creation. Samples are obtained via Sequential Importance Sampling (SIR) as described.
The sequential estimation of the subsequent distribution, \( p(x_t | y_{1:t}) \), is then found as a solution to the Chapman Kolmogorov integral equation:

\[
p(x_t | y_{1:t}) \sim p(y_t | x_t) \int p(x_t | x_{t-1}) \frac{p(x_{t-1} | y_{1:t-1})}{p(x_{t-1} | y_{1:t-1})} dx_{t-1}
\]  \( (1) \)

which, in practice, is solved by Monte Carlo integration. The SMC method is a sequential version of the MCMC algorithm, where the present time step only depends upon the previous time step and the evolution dynamics is restricted with a carefully crafted likelihood condition.

Intuitively, the key idea of the method is that we can approximate the filtering density by a collection of \( N \) weighted particles within the state space of the system. These so-called particles represent observation samplings of a given quantity, probability distribution, or process, which in turn give rise to probabilistic weighting, used for making inference concerning the hidden state of the system from a chosen likelihood criteria. Upon each sequential time update of the system, an iterative sampling/ resampling process is performed in order to adaptively mold the obtained posterior distribution at time \( t \) into one that best represents the system before the next update. This operation has a crucial effect upon the convergence of the method.

A more mathematical description of the SMC process is as follows. For a given input \( p(x_{t-1} | y_{1:t-1}) \) (the state at time \( t - 1 \)), we are interested in finding the output \( p(x_t | y_{1:t}) \) (the state at time \( t \)). In practice, this algorithm proceeds with the following iterative steps:

1. **Prediction by sampling the state space:** While a formal solution is found by solving Equation 1, we replace the integrals with Monte Carlo sums by using importance sampling, so that:

\[
\int p(x_t | x_{t-1}) p(x_{t-1} | y_{1:t-1}) dx_{t-1} \rightarrow \sum_j w_{t-1}^j p(x_t | x_{t-1}^j)
\]

where \( j \) represents each random samples, \( w_{t-1}^j \) is the likelihood, or weight, of each sample (or particle) at time \( t - 1 \), and \( x_{t-1}^j \) is the hidden state which was inferred at time \( t - 1 \).

2. **Bayes update for calculation of weights** The weights at time \( t \) are found by applying Bayes rule.

\[
p(x_t | y_{1:t}) = \frac{p(y_t | x_t)p(x_t | y_{1:t-1})}{\int p(y_t | x_t)p(x_t | y_{1:t-1}) dx_t} \rightarrow w_t \approx \frac{w_{t-1}^{(i)} p(y_t | x_t^{(i)}) p(x_t | x_{t-1}^{(i)})}{\sum p(x_t^{(i)} | x_{t-1}^{(i)}, y_{1:t})} \quad \quad (2)
\]

where the equation for \( w_t \) is the weight obtained by replacing the integrals with Monte Carlo sums. The result is the output \( p(x_t | y_{1:t}) \), which gives the probability that an object is at state \( x_t \), or equivalently expressed through the weight. The update step is equivalent to calculating weights obtained by evaluating the present samples at time \( t \) against a likelihood metric, which in this case are histogram comparisons between each sample \( s_i \) and the reference histogram distribution \( \pi_e \), obtained at the start of the trajectory. The update of the \( k \)-th weight is calculated as: These weights are then used for obtaining a prediction of the state dynamics through a model, which in our case is a second order autoregressive model so that the system moves recursively through time.
3. Thus, we propagate the set \( \{ x^{(i)}_{t-1}, w^{(i)}_{t-1} \} \) recursively through the system given some dynamical function: \( x^i_t = f(x^i_{t-1}) \) in order to obtain new samples and associated weights at the next timestep.

4. **re-sampling**: perform a mutation/selection or a redistribution/re-sampling to maximize the likelihoods; this is done by multiplying particles \( x^{(i)}_t \) with large weights \( w^{(i)}_t \) and discard ones with small weights.

A schematic of the procedure for implementing the algorithm is shown in Figure 2 which illustrates the time evolution of a cell (object to track) from time \( t \) to \( t+1 \) (and positions \( x_t \) to \( x_{t+1} \)). In Figure 2 the squares represent individual particles, which are samples of the underlying image pixels; these are small region of the image (e.g. \( 16 \times 16 \)) which are used for obtaining the color histogram. The likelihood of the \( j \)th particle (proportional to the weight) \( w_j \sim p(\gamma_j) \), is calculated with the Poisson distribution, where \( w_j \sim A_j \exp(-\lambda D(h_j, h') \) where \( A_j \) is a normalization constant, \( D(h_j, h_{ref}) \) is a distance metric between the two histograms \( h_j \) and \( h_{ref} \) obtained between the \( j \)th particle (sampled regions of the image) and a reference. The most probable particles have the largest weights and are indicated in Figure 2 with grey filled squares.

![Figure 2: The SMC method for tracking objects, illustrating resampling and update steps.](image)

After a resampling procedure (see below), the particle positions are updated from time \( t \) to \( t+1 \) through a dynamic transformation \( x_{t+1} = T(x_t, x_{t-1}, N) \) that consists of a deterministic part (calculating the position from velocity and previous position) and a stochastic part, that uses a Gaussian distribution centered around the deterministic prediction:

\[
x_{t+1} = T(x_t, x_{t-1}, N) = \underbrace{a_1 x_t + a_2 x_{t-1}}_{\text{deterministic prediction}} + \underbrace{b N(\mu, \sigma; \theta)}_{\text{stochastic term}}
\]

where the constants \( a_1, a_2, \) and \( b \) are free parameters determined empirically, the Gaussian \( N \) provides a normal sampling distribution depending upon an offset \( \mu \) (which is normally set to zero), the width \( \sigma \), and a parameter \( \theta \) that uses the local velocity vector to influence the stochastic sampling within a directed cone along the direction of motion.

**Resampling** As mentioned, the resampling step is critical to the convergence of the SMC method, since it conditions the prior probability for the next updated state. Indeed, present
Subsequently, given by $C$ (area constraints). We find the weights assigned to each particle between node pairs (two point constraints) and by conservation of the closed area within nodes about the underlying pixels. The nodes of a cluster are constrained by maximizing the weights of particle $j$ at time $t$, obtained from square image regions containing information about the underlying pixels. The nodes of a cluster are constrained by maximizing the weights between node pairs (two point constraints) and by conservation of the closed area within nodes (area constraints). We find the weights assigned to each particle $\gamma_j$ in each node by taking a product of all the constraints: $w^k_j = p_{\text{model}} \prod_{j \in \text{constraints}} w^k_j$, so

$$w^k_j = w^{k}_{\text{pixel}} \cdot \left[ f_{\text{pairs}} w^{k}_{\text{pairs}} \right] \cdot \left[ f_{\text{area}} w^{k}_{\text{area}} \right]$$

where the constants $w^{k}_{\text{pixel}}$ are the normalized particle weights obtained from the likelihood criteria, $f_{\text{pairs}}$ and $f_{\text{area}}$ are free parameters of the model, which are set to give the relative strength between the two types of constraints and whose values are chosen experimentally.

### Two point constraints

The two point constraints are meant to conserve the approximate spatial distance between nodes within a cluster. Since each node consists of $m$ non-connected particles, the constraint condition must be applied pairwise to all combination of particles between each node. Thus, for a given cluster $C_j$, we find all pairwise distances from the location of the $j$th particle in node $k$, $x(\gamma^k_j)$, and the location of the $i$th particle in node $n$, $x(\gamma^m_i)$. From this distance, we can derive a likelihood weights, $w_{i,j}^{(k,n)}$ that are sharply peaked around reference distance $\Delta_{\text{ref}}$, specified as an initial condition, written as follows:

$$w_{i,j}^{(k,n)} = \exp \left( \frac{(\Delta_{\text{ref}} - (x^k_j - x^m_i))^2}{\sigma_{\text{pair}}} \right)$$

(4)

Subsequently, this weight is normalized and multiplied to each pair of particles. An important aspect of the algorithm is that these node constraints are soft, since they are applied through the total probabilistic weights, and therefore distances between nodes that are smaller or larger than the reference distance are still valid, albeit with smaller probability.
**Area constraints** An important assumption is that the volume of the cell remains constant during the trajectory. For the limited case where we track cells within one of the planes from the z-stack over time, the area will not remain constant since we are slicing the cell. The area can be constrained by multiplying a weighting factor to all sets of particles from each node in the cluster, in a similar way to that assigned with the two-point constraint. In this case, the area constraint likelihood function is highly peaked around a reference area $A_{\text{ref}}$, obtained as an initial condition. Then, the areas of all possible sets of vertices from the $m$ particles from each node $k$ of a cluster must be calculated in order to compare against $A_{\text{ref}}$. We denote the set of all particle indices from node $k$ and particle $j$ from each cluster that form an area $A_i$ as: $S = \{(k,j)_i\}$, where the $i$ are enumerated by forming the Cartesian product. Thus, the weights for each of the indices in $S$ are found in the following manner:

$$w_{\text{area}}(S) = \exp\left(\frac{(A_i - A_{\text{ref}})^2}{\sigma_{\text{area}}}\right)$$

which must be calculated for all sets within the Cartesian product.

From a practical point of view, there are two methods that we have implemented:

- **brute force**: A full enumeration of all possible areas whose vertices are the particles from each node are found from the Cartesian product. For a small number of particles and nodes, such an enumeration is feasible. However, for even modest sized problems such an approach, although implemented, is prohibitively expensive in practice.

- **Monte Carlo sampling from maximal two-point constraints**: We have developed an alternative, more efficient method for obtaining an approximate maximum weights. In this algorithm, the set of candidate particles in each node is pre-conditioned by two point constraints, under the assumption that if the lengths are close to the reference, then the area will be more likely conserved. This reduced set is used for sampling with a Monte Carlo algorithm and therefore only requires a fixed number of operations. This method works well since many weights that are calculated with the brute force Cartesian product may be only slightly better than those found with the Monte Carlo. Also, we are not seeking a global maximum solution for the weights, but a solution that is close to the maximum.

In either case, the maximum solution is highly peaked, with likely solutions lying close to the maximum.

**Cluster Dynamics** In order to illustrate how a constrained SMC cluster is updated, Figure 3 shows the time evolution of a cluster, positioned over a tracked object, from time $t$ to $t+1$ (and positions $x_t$ to $x_{t+1}$). In this figure, the nodes of the cluster consist of the cloud of $m$ particles together with their positions (and other attributes), connected through probabilistic weights. Each node of Figure 3 are represented graphically by (dashed) diamonds. As described previously, the SMC algorithm advances the particle state (position and weights) from from $t$ to $t+1$ through a transformation $x_{t+1} = T(x_t, x_{t-1}, \mathcal{N})$ that depends upon previous states through a second order recursion, together with a stochastic sampling of a Gaussian distribution.
Cluster interactions and cell merge problem  

As mentioned previously, our technique seeks to succeed for cases of cell merging where other available methods fail. Since clusters will maintain an area, these clusters can be made to repel (or non-overlap) each other, and therefore maintain the initial cell identity.

The problem with existing methods is that when cells briefly stick together (or when the trajectories cross) in the course of their movement, information about the cell boundaries is lost in the images at the point of contact. Thus, the algorithms may produce a mislabelling error due to the ambiguity that results, thereby leading to tracking errors. If the cluster dynamics and the interaction conditions of clusters is included in the tracking algorithm, these ambiguities can be reduced when cells undergo an inelastic collision and stick to each other. In this way, cluster-cluster interactions can be made to mimic the actual cells: first, they conserve area while still changing shape and second, a repulsive force can be used to represent the fact that the membrane cannot collapse yet maintaining the cells as separate entities.

A schematic of the situation with two cells that merge, together with the constrained SMC cluster tracking is shown in Figure 4. In this figure, different time points are shown for two approaching cells, and are tracked with clusters $C_1$ and $C_2$. In a low-contrast microscopy image, the border between the two cells would no longer act as a reference for an algorithm, and an ambiguity could arise as to the location of each cell. By using clusters, this situation can be remedied by the fact that they will experience a short range repulsive force giving rise to a small outward pressure, and are not allowed to overlap. Combined with predicted velocity direction, the individual cells can still be identified despite the fact that no border information is available.

Our model for the repulsive force between two clusters $C_i$ and $C_j$ is given as a function of the distance between their centers of mass: $\rho_{i,j} = (x_{cm}^i - x_{cm}^j)$. Thus, the central repulsive force vector acts upon each cluster with the vector field:

$$ f_r(C_i, C_j) = \begin{cases} \alpha_r / (\Gamma - \beta \rho) & \text{if } \rho < \rho_{min} \\ 0 & \text{if } \rho \geq \rho_{min} \end{cases} $$

where $\alpha_r$, $\Gamma$, $\beta$ and $\rho_{min}$ are free parameters that depend upon the object sizes, and have been determined empirically. Since the force is designed to short range, outside the minimum distance, $\rho_{min}$, the force should be negligible, and therefore no repulsive correction is applied.
3 Algorithms and Software Implementation

The software tool we have developed that implements the constrained SMC cluster algorithms described in the previous section. Our software tool is written in Python and relies upon Numpy, Scipy, and OpenCV[24], which is a well known computer vision library. The Python language, and associated modules, provides a powerful platform for rapid code development within an easily extendible environment. By integrating powerful python modules: Numpy and Scipy, OpenCV and also Matplotlib, complex computer vision tasks can be implemented efficiently. Due to the fact that our method relies upon Monte Carlo, we are presently implementing sections to run with OpenCL in order to take advantage of graphics processing units (GPU) computing, gaining on average a factor 30×.

Figure 5 is a highly simplified flow diagram of our software implementation for tracking cells. The input step and initial image processing can treat complex data sets obtained from microscopy experiments as well as simple video files. Our implementation requires users to initiate the first frame by selecting the cells to be tracked. Given this initial condition, the program flow consists in performing the SMC updates, applying the constraints in order to form clusters, and output of particle track information, until the last frame is reached.

A more detailed description of the algorithm is as follows:

**Step 1** *Input data:* capture video files or experimental data in the form of a directory of files; For 3-dimensional data each layer must be aligned with the underlying layer. At present, the algorithm contains processing for single layers of the z-stack.

**Step 2** *Image filtering/treatment:* This step consists of standard image analysis techniques such as smoothing and thresholding in order to produce more reliable cells for tracking. Real
Step 3 Object Selection and initial conditions: The cells to be tracked are selected interactively by the user. The user is able to select the number of clusters, the number of nodes in the cluster and the total number of particles. Other free parameters can be fixed at this initialization stage.

Step 4 Main loop of tracking: The main loop consists of running through the $N_F$ frames of the video sequence. In its simplest form, at $t = 0$, that is for the first frame, we establish the $k = 1 \cdots N_p$ cells to be tracked by interactively defining the particle nodes as previously described. Once the initialization is made, reference histogram of each node $k$ is obtained, by calculating histograms from the HSV color-space.

Step 4B Sampling, Dynamics and weights:

- For $t > 0$ and for each of the clusters, we loop over all the nodes performing the SMC-based state transitions, as described in the previous section. Also, from the details of the likelihood conditions, the weights are calculated from the underlying pixel information.
- For subsequent frames ($t > 0$), we must now create a swarm of $j = 1 \cdots N_p$ particles, sampled around the center of the particle node $\rho_k(t)$. The particles of node $k$ at time $t$ are denoted $\gamma_j^k(t)$.
- For each of the $j$ particles, we calculate the transition probabilities $p(x_t|x_{t-1})$, by using a simple state dynamics prediction model in the form of a second order autoregressive model. Thus, the trajectory at $\mathbf{x}_{t+1}$ is obtained by including information from previous states together with a stochastic sampling, as described in Equation 3.
- The next critical step in the particle filter algorithm is to calculate weights of each particle at each time $t$. Thus, $w_j^k(t)$ is the weight for the $j$–th particle in node $k$, at time $t$, and is obtained by calculating the likelihood of the sample from the underlying image. The likelihood condition is found by performing a comparison of the histogram for each $p_j^k(t)$ with the reference histogram $h_k(t_0)$, thereby obtaining the maximum likelihood estimate, $\exp[-\lambda D^2(h,h^*)]$, where $D$ depends upon the specific distance metric used, and $h$ and $h^*$ are the two histograms to be compared. The histogram comparisons are performed with a Bhattacharjee distance metric.

Step 5 Apply the two point constraints: As described in the previous section, we form all unique pairwise combinations between the particles in each node of the cluster and calculate the weight, $w_j^k$ as in Equation 4.

Step 6 Apply the Area constraint: In order to preserve the area of the clusters that track the cells, we apply the area constraint in order to obtain the weights, $w_{\text{area}}(S)$, by using a Monte Carlo sampling from the maximal set of two-point constraints.

Step 7 Resampling: The resampling step is critical to the SMC particle filter sampling, which is to normalize the particle weights, thereby obtaining $\tilde{w}_j^k(t)$, which is subsequently used for resampling the particle distributions for the next timesteps, $p_j^k(t+1) \leftarrow \tilde{p}_j^k(t)$.
Step 8 **Repulsive correction:** The short range, pairwise repulsion is applied directly to the dynamic variables $x_t$ of all particles in the cluster $C_k$ and $C_j$, whose centers form a distance $\rho_{k,j} < \rho_{\min}$. As described in the previous section, this is a hallmark of our constrained SMC cluster algorithm for dealing with the ambiguities which arise during cell merging and crossings.

Step 9 **Track output:** Finish tracking and write out tracks.

## 4 Two-photon Data Analysis

Our two-photon data was collected from lymph nodes located in the back part of the hind limbs of mice, close to the knee height. These organs are called popliteal lymph nodes. Animals were deeply anesthetized and submitted to a small surgery to expose this organ. After exposure, the whole animal was placed on top of a heating pad ($37^\circ$ C) and the heating reached the organ throughout conduction by an aluminum foil connected to the heating pad. A saline physiological buffer was placed on top of this region to keep the organ moist. Each animal also received an intravenous injection of Rhodamine B isothiocyanate-Dextran (Cat.# R9379, Sigma-Aldrich, USA) to allow blood vessel location and flow, thus helping to locate the relative position of tissue T cells. These animals also express GFP under one specific gene promoter (Foxp3 gene promoter [25]) and, therefore, T cells called regulatory T cells (Tregs) can be observed by fluorescence microscopy. Every 40 min, a new boost with anesthetics was provided to avoid any animal suffering. The whole preparation was transferred to our two-photon microscope (Prairie Technologies, Inc., USA) where the images were acquired.

![Figure 6](http://journal.imbio.de)  
**Figure 6:** The $z$-stack representation and corresponding images for the color channel showing blood vessels.

Once the region of interest was located, we started time-lapse imaging acquisition. In our case, a typical 5D ($x, y, z, t,$ and color) acquisition protocol consists of acquiring sequential images of a 50 $\mu$m-depth tissue volume, divided into 4.0 $\mu$m $z$-steps. Each volume takes around 30 seconds to be acquired. Therefore, 60 acquisitions will perform around 30 min of image acquisition. An illustration of the two separate channels from parts of the data stacks are shown in Figures 6 and 7.
5 Results and Discussion

We used our software to analyze our own data from the in-vivo two photon data, described in the previous section as well as 2D data from videos from a series of public datasets.

**Two-photon Treg Data**  The dataset containing Treg cells from the two-photon data collected from lymph nodes was analyzed with our software. Because the experimental setup is complicated by the fact that heat needs to be delivered directly to the exposed organ of the animal, there is considerable global movement in all directions $x, y, z$, (of approximately $5 \mu$) due to the thermal expansion of the heating coil. This motion directly effects the motility extraction and needs to be removed prior to tracking cells. For this, we developed an image registration algorithm that aligns the $z$-stack of the fluorescent channel showing the blood vessels. These are large structures that provide an approximate rigid background as compared to the T cell motility.

Once the global background motion was subtracted, we extracted a single layer of the $z$-stack.
from the fluorescent Foxp3 channel for all time points. For this image sequence, we performed standard image analysis including image filtering, thresholding, and contour extraction in order to segment the T-cells from the background. We then applied our cluster SMC algorithm by interactively selecting groups of cells to be tracked. In order to demonstrate functioning of our algorithm, Figure 8 shows a sequence of image frames obtained from the output of our tracking program for the problem of cell joining. As can be seen in the figure, the SMC clusters approximately maintain their areas and due to the weak repulsive force, they continue to properly identify the individual cells despite the fact that any reference to the cell membrane is not discernible from the microscopy images.

![Image](image.png)

(a) Tracking two B-cells in microscopy video sequence

![Image](image.png)

(b) Representative position and velocity information

Figure 9: Image time sequences and analysis: (a) Image sequences showing the tracks overlayed on the moving cells. (b) Example visualization methods for superposed tracks: (left panel) center shifted tracks; (middle panel) unshifted tracks; and (right panel) time-average track directions per bin.

**Public Datasets** In order to further show the basic functionality of our software, we have initially used example videos provided in the supplementary material from recent articles on the affinity maturation process from Allen [10], Schwickert [17] and Hauser [26]. While this public dataset contains only a curated subset of the original frames, thus limiting our ability to demonstrate the full capability of the software, we can show preliminary results. Thus, as an example of tracking several cells simultaneously Figure 9A shows three image frames, obtained from reference [26], superposed with SMC obtained tracks from our software.

From these datasets, there are two types of results which are of interest. First, we are interested in the accumulated error and instantaneous displacement of the SMC obtained track as compared to the true track obtained manually (possibly with Retrack). We found that accumulated root square error for each track was less than 10% and the deviation was consistently less than 5%.
The other type of results that we can demonstrate are various representations useful for immunological models of germinal centers. In particular, from the raw motility coordinates along each cell track, our software can produce the following representations: center shifted tracks, unshifted tracks, and time-averaged track direction. In Figure 9B, the track information corresponding to the experiment in Figure 9A is extracted and position and velocity information is plotted for a finite number of cells. Similar plots can be found in the references [27] for demonstrating that a further analysis yields information such as directed motion, or flux from one region to another. Given the fact that our system extracts positions and movement of individual cells, it is possible to reconstruct several other important data representations, as suggested in the recent review by Beltman et al. [28].

6 Conclusions

The constrained SMC cluster method described in this paper represents a novel solution for the complex problem of tracking low-contrast microscopy imaging of cells. It has the potential of overcoming several problems associated with present methods applied to this issue. The strength of our approach is that it is specifically designed to treat the ambiguous cases of cell merging and crossing by incorporating dynamics and weakly interacting (nonoverlapping) SMC clusters. Another advantage is that the method extends to 3-dimensions in order to treat the entire z-stack over time and get a larger range of motion of the cells during the immune response.

More comprehensive demonstrations of our software can be found on the application website [29].

Acknowledgements  DO acknowledges the support of grant 08SIN002206PR (Xunta de Galicia, Spain). JF acknowledges the support of grants SAF2007-63152 and HP2008-0018 (MEC, Spain), and PIRSES-GA-2008-230665 (7th FP, EC).

References


[29] http://milegroup.net/demos/SMCcelltracking