Differentiable Event Stream Simulator for Non-Rigid 3D Tracking

Jalees Nehvi[1,2] Vladislav Golyanik[2] Franziska Mueller[3]
Hans-Peter Seidel[2] Mohamed Elgharib[2] Christian Theobalt[2]

[1]Saarland University [2]MPI for Informatics, SIC [3]Google Inc.

Abstract

This paper introduces the first differentiable simulator of event streams, i.e., streams of asynchronous brightness change signals recorded by event cameras. Our differentiable simulator enables non-rigid 3D tracking of deformable objects (such as human hands, isometric surfaces and general watertight meshes) from event streams by leveraging an analysis-by-synthesis principle. So far, event-based tracking and reconstruction of non-rigid objects in 3D, like hands and body, has been either tackled using explicit event trajectories or large-scale datasets. In contrast, our method does not require any such processing or data, and can be readily applied to incoming event streams. We show the effectiveness of our approach for various types of non-rigid objects and compare to existing methods for non-rigid 3D tracking. In our experiments, the proposed energy-based formulations outperform competing RGB-based methods in terms of 3D errors. The source code and the new data are publicly available[1].

1. Introduction

Template-based non-rigid 3D tracking using a single monocular RGB camera is a well-studied and challenging problem [16, 13, 30], due to the ambiguities associated with the monocular setting (*i.e.,* occlusions, lost depth information and many-to-one mapping from the 3D space to the 2D image plane). In contrast, 3D tracking non-rigidly deforming shapes from a single monocular event camera is an emerging research field. To the best of our knowledge, *there has been no method shown in the literature so far which can track general objects in 3D from a single event stream and without prior knowledge of the observed object class* (in contrast to EventCap [29] which assumes human bodies).

While conventional cameras record images synchronously, *i.e.,* at regular intervals (*e.g.,* every 33 ms) and full or partial resolution of the physical pixel matrix, event cameras react to brightness changes asynchronously in space and time. Each signal, called *an event*, reports a change of the per-pixel brightness by a certain threshold; it contains the pixel coordinates of the event occurrence, a timestamp and polarity. Thanks to the asynchronous operational principle, timestamp resolution of modern event cameras reaches $1\ \mu s$. Polarity is a binary flag showing whether the brightness at a given pixel has increased or decreased. Advantages of event cameras include high dynamic range, ultra-high temporal resolution and lower data throughput than the throughput of high-speed RGB cameras recording fast moving and deforming objects.

Thus, compared to the monocular RGB-based case, 3D tracking from an event camera poses **additional challenges**. Not only because the colour information is lost but also due to the different data modality, the existing methods for monocular template-based 3D reconstruction cannot be applied to event streams. While some principles developed over the decades for the RGB-based setting can be extrapolated to event streams in order to make the problem well-posed (*e.g.,* the assumptions of isometric deformations and volume preservation), entirely new techniques are required to guide the reconstruction. Talking about energy-based methods, this means that the data term and geometric regularisers have to be questioned and re-invented for the new and more challenging setting and data modality.

This paper proposes *a new differentiable event stream simulator for problems on event streams involving non-rigidly deforming objects*. The core idea of our technique is to correlate a synthetically generated event stream pattern with an observed one. *We combine computer graphics rendering techniques with the event stream formation model in a differentiable way and obtain a novel analysis-by-synthesis approach.* Assuming the initial 3D shape state is known and accurately projects to the image plane—which is the assumption of many monocular template-based non-rigid 3D tracking methods [30]—we iteratively update the 3D geometry until the induced event stream pattern strongly correlates with the observed event stream pattern.

[1] http://gvv.mpi-inf.mpg.de/projects/Event-based_Non-rigid_3D_Tracking

One of the advantages of such an approach is that it is correspondence-free, *i.e.,* it does not rely on spatio-temporal event association, unlike [29]. Second, it works entirely in the event space and does not require complementary greyscale images, unlike [29]. Third, our component can be readily used in modern neural approaches and, potentially, other problems involving event streams. To summarise, the main **contributions** of this paper are as follows:

- The first differentiable event stream simulator for problems on event streams using analysis-by-synthesis and event stream correlation approach.
- The first energy-based method for template-based tracking of deformable surfaces in 3D from a single event stream relying on our differentiable event stream simulator. We show variants of its energy function for parametric shapes and general meshes. Our method is fully unsupervised and does not require training data.

In the experiments with three types of non-rigid objects (hands, thin surfaces and watertight meshes), we demonstrate that our method reaches 3D tracking accuracy comparable to what is achievable by modern RGB-based methods. The proposed approach is more accurate at tracking fast deformations, which cause motion blur when recorded with a conventional RGB camera. Moreover, in several cases, our event-based formulation also outperforms RGB-based methods in the case of moderately evolving surfaces.

2. Related Work

Monocular 3D Non-Rigid Tracking. Methods for 3D reconstruction of non-rigid surfaces from monocular videos can be classified into non-rigid structure from motion (NRSfM) and template-based approaches. NRSfM relies on correspondences across the input views relative to a single keyframe, which are factorised into camera poses and non-rigid shapes for every input frame. Whereas earlier methods focused on the factorisation of sparse tracked keypoints [1, 27], recent methods support dense optical flow as input [5, 12, 8, 26]. Template-based methods assume that an accurate initial 3D state for one sequence frame is known [9, 23, 16, 14, 13, 30]. Several recent methods encode templates in weights of neural networks, which are trained to regress 3D surfaces directly from 2D images [25, 28].

Event-Based 3D Reconstruction. Since event cameras became accessible for the broad research community, they were applied to various low- and mid-level computer vision tasks [6]. Several methods have been proposed for 3D reconstruction and tracking of rigid objects in 3D [24, 10, 20, 18]. In contrast, 3D reconstruction of non-rigid objects from a single event stream remains a starkly underexplored problem in the literature. Calabrese *et al.* [3] propose a method for 3D human pose estimation from multiple synchronous event streams. EventCap of Xu *et al.* [29] is a hybrid approach that relies on deblurred greyscale images recorded at usual frame rates and an event stream from DAVIS240C to track a rigged 3D model of an actor.

The recent work of Rudnev *et al.* [22] proposes EventHands, *i.e.,* a neural method for 3D hand reconstruction from a single event stream. In contrast to EventHands, our method does not require large corpora of training data, and works for general non-rigid objects with hands only being one example object. All in all, we propose a new way of solving tracking problems from a single event stream in an analysis-by-synthesis fashion, which can be potentially used for other problems than 3D tracking in future.

Our event correlation approach relates to the method of Bryner *et al.* [2], who determine an event camera pose with respect to a rigid 3D reconstruction of an environment. Their objective function minimises the difference between the measured (*i.e.,* obtained by integration of real events) and predicted (*i.e.,* obtained by rendering the 3D map) intensity images. In contrast, we compare event frames, *i.e.,* observed and predicted, and support deformable objects.

While there are existing event stream simulators such as ESIM [19], we implement own lightweight component that is fully differentiable and easily customisable for target non-rigid 3D tracking and reconstruction applications. Note that we do simulate individual events, which are then accumulated in windows for comparisons. On the other hand—and as our experiments show—the implemented level of event synthesis fidelity is sufficient for our target applications.

3. Event Generation Model and Event Frame

Event Generation Model. In contrast to RGB cameras which record absolute brightness, event cameras record relative brightness changes in form of asynchronous events. At a given pixel $\mathbf{x}$, an event is triggered if the absolute value of the difference between the incoming brightness received at that pixel and the current brightness stored at that pixel, exceeds a certain threshold called the *contrast sensitivity* of the event camera. The pixels are triggered independently, resulting in an asynchronous stream of events, when

$$|\mathcal{L}(\mathbf{x}, t) - \mathcal{L}(\mathbf{x}, t - \Delta t)| \geq C \,, \tag{1}$$

where $\mathcal{L}$ represents the brightness, t represents the current timestamp and C represents the event camera threshold.

Event Frame. An event is an indicator of brightness change at a given pixel at a particular instant of time. It is usually represented as a tuple $e = (x, y, t, p)$, where x and y denote the pixel coordinates, t the timestamp and $p \in \{-1, 1\}$ the polarity which indicates whether there has been a positive or negative change in brightness. Since individual events carry little information, we accumulate several events over a span of time. Let $\mathcal{S} = \{e_i = (x_i, y_i, t_i, p_i)\}$ be the set of events that occurred within a time window T. We consolidate these

events into a single *event frame*

$$\mathcal{E}(\mathbf{x}) = \sum_{e_i \in \mathcal{S}} p_i \, \delta((x_i, y_i) - \mathbf{x}) \, , \tag{2}$$

where δ is the impulse function. T is set such that there is enough information per event frame and the high temporal resolution property of the event camera is not compromised. The event frames are normalised in the range $[-1, 1]$.

4. Overview

Our goal is 3D tracking of non-rigid objects from event streams. Since our differentiable event simulator is independent of the exact parameterisation of the object of interest, it can be used with parametric 3D morphable models as well as general 3D meshes. For each event frame, we aim to estimate the current 3D geometry and global rigid transform, consisting of translation t and rotation R, such that the generated event frame is aligned with the incoming event frame. For a parametric model (*e.g.*, hands [21, 17]), the 3D geometry is controlled by the pose parameters θ, yielding the set of parameters $\Theta = (\theta, t, R)$. For general 3D meshes, we assume the availability of a template mesh and express the 3D geometry as displacement field of the vertices. In this scenario, the set of parameters we optimise is $\Theta = (\mathbf{V}, t, R)$, where $\mathbf{V} = \{v_i\}_{i=1}^{N}$ are the 3D coordinates of mesh vertices and N is the total number of vertices.

Given the input event frames and the parametric model or a mesh, we start from an initial parameter set Θ and render an image of the mesh. Our goal is to find a parameter set such that the events generated by an *event generation model* (see Sec. 3) match the incoming events. We optimise the unknown parameters by minimising the objective function in a frame-wise manner, see (5) and (10). After we have found the optimised parameters for the first event frame, we use them as an initialisation for the next event frame and proceed in a similar manner for subsequent frames.

In our experiments, we use the DAVIS 240C event camera which captures both intensity images as well as event streams of 240×180 pixels with temporal resolution in the microsecond range. The intrinsic camera parameters are known. Note that our method is purely based on the asynchronous event stream input, and hence we do not use the complementary greyscale images of DAVIS 240C.

5. Method

An overview of our framework is given in Fig. 1. We want to find the optimal parameters Θ^* by comparing the event frame generated with the current parameter hypothesis Θ to the input event frame. First, we use a differentiable renderer to render two images using the meshes from the current and previous timestamps. We then calculate the per-pixel differences and threshold them according to the event generation model. To ensure the differentiability of the thresholding process, we use a differentiable approximation of the staircase function. Finally, the generated events are used in objective function terms which compare the incoming events with the generated events (*i.e.*, by means of signal correlation).

5.1. Differentiable Event Simulator

The main component of our framework is the differentiable event simulator which generates the event frame corresponding to the current parameter hypothesis Θ. This synthesised event frame is later used for optimising an analysis-by-synthesis objective function. To obtain the event frame, we render two greyscale images, one corresponding to the mesh at the previous time stamp and one corresponding to the mesh at the current parameter hypothesis Θ. We make use of differentiable rendering such that the images are differentiable with respect to the parameters Θ controlling the mesh deformations. We normalise the rendered images in the range $[0, 1]$, which is based on an empirical observation and leads to stable gradients. We assume that the object's surface is Lambertian and the lighting conditions are constant. The rendering is followed by the subtraction of the previously rendered image from the current one. Finally, the difference image is thresholded using the smooth thresholding function discussed below, resulting in a generated event frame.

Thresholding Function. We approximate the inequality operation in the event generation model in the form of a smooth threshold function which is a trade-off between the hyperbolic tangent function and the staircase function:

$$g(x) = \left(\frac{x + \epsilon}{|x| + \epsilon} \right) \left(\frac{1}{1 + e^{-w|x| + wC}} \right), \tag{3}$$

where x represents the difference image, w denotes the weight that controls the smoothness of the curve, C represents the threshold or contrast sensitivity of the event camera and ϵ is the tolerance which ensures stable gradients.

In general, all the aforementioned steps involved in event frame generation can be mathematically formulated as

$$\mathbf{e}(\Theta^{t-1}, \Theta^t) = g(\Pi(\mathcal{M}(\Theta^t)) - \Pi(\mathcal{M}(\Theta^{t-1}))), \tag{4}$$

where $\mathbf{e}$ is the generated event frame; Θ^t and Θ^{t-1} represent the current parameter set and the parameters estimated for the last frame, respectively; $\mathcal{M}$ is the parametric model or generic mesh; Π denotes the image rendering operation and g is the smooth threshold function defined in (3). The generated and input event frames are then fed into an optimisation framework minimising objective functions—which we describe in Secs. 5.2 and 5.3—to estimate the unknown optimal parameters Θ^* in a frame-wise manner.

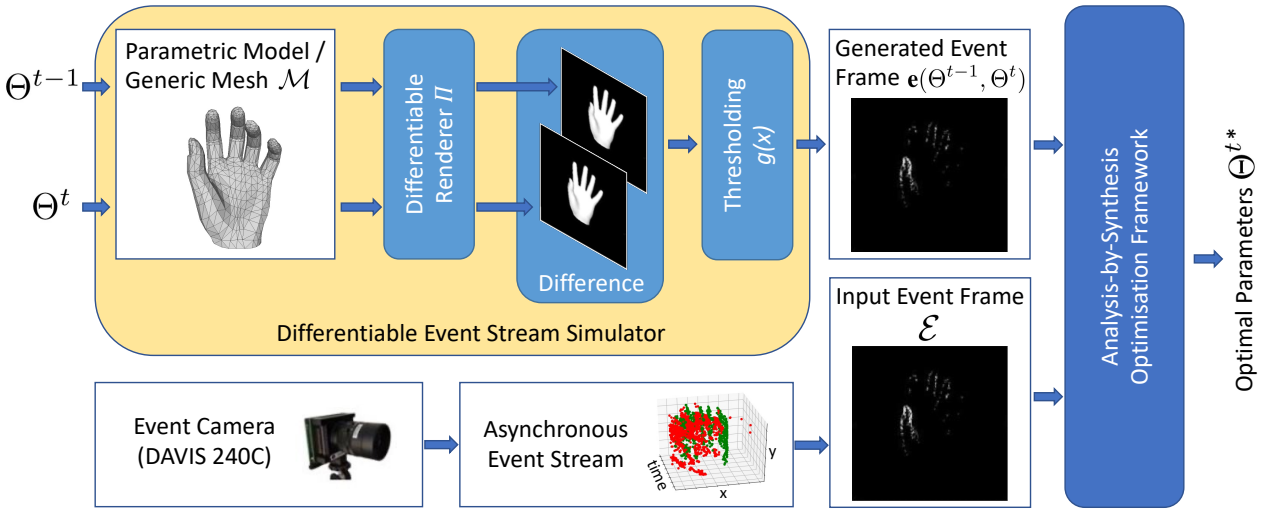

Figure 1. **Our non-rigid 3D tracking framework.** We propose a new differentiable event simulator to generate event frames for the current pose hypothesis Θ. We then optimise the parameters Θ by comparing the generated event frame to the input event frame in an analysis-by-synthesis optimisation framework. The positive and negative events of the input stream are shown in green and red, respectively.

5.2. Objective Function: Parametric Model

In the case of a parametric model like hands (we use MANO [21]), the objective comprises of two main terms, *i.e.,* a data term and a regularisation term:

$$E(\Theta) = E_{data}(\Theta) + \lambda E_{reg}(\Theta)\,, \tag{5}$$

where λ is a regularisation hyperparameter and $\Theta = (\theta, t, R)$ (see Sec. 4) represents the set of all parameters, consisting of model parameters for pose θ and the rigid transform defined by translation t and rotation R.

5.2.1 Data Term

To account both for the occurred and missing changes in the observed 3D shape, we split the data term into an *event* and a *no-event* terms:

$$E_{data}(\Theta) = E_{event}(\Theta) + \lambda_1 E_{no\text{-}event}(\Theta)\,, \tag{6}$$

where λ_1 balances the two data terms.

Event Term: The event term penalises the difference between the input event frame $\mathcal{E}$ and the generated event frame (Eq. 4), at only those pixel locations where an event is present in the input event frame.

$$E_{event}(\Theta) = \sum_{\mathbf{x}\in\Omega} \big(\gamma(\mathbf{x})\big(\mathcal{E}(\mathbf{x}) - \mathbf{e}(\Theta^{t-1}, \Theta)(\mathbf{x})\big)\big)^2\,, \tag{7}$$

where Ω denotes the spatial image domain and $\gamma(\cdot)$ is an indicator function that yields 1 if an event occurred at pixel $\mathbf{x}$ in the input event frame and 0 otherwise.

No-Event Term: On the contrary, the no-event term penalises any non-zero value present in the generated event frame at the corresponding locations in the input event frame where an event is absent:

$$E_{no\text{-}event}(\Theta) = \sum_{\mathbf{x}\in\Omega} \big(\bar{\gamma}(\mathbf{x})(\mathbf{e}(\Theta^{t-1}, \Theta)(\mathbf{x}))\big)^2\,, \tag{8}$$

where $\bar{\gamma}(\mathbf{x}) = 1 - \gamma(\mathbf{x})$ indicates pixels where no event occurred in the input event frame. Note that without $E_{no\text{-}event}$, the tracked 3D shape would not be sufficiently constrained.

5.2.2 Regularisation Term

The regularisation term ensures temporal smoothness by penalising large parameter changes between the current and the previous event frame. This helps to reduce flickering of the estimated mesh between adjacent frames:

$$E_{reg}(\Theta) = \|\Theta - \Theta^{t-1}\|_2^2, \tag{9}$$

where Θ is the current parameter set and Θ^{t-1} is the parameter set estimated for the previous frame.

5.3. Objective Function: Meshes

Instead of a parametric model, here we are provided with a generic mesh parameterised by $\Theta = (\mathbf{V}, t, R)$ where $\mathbf{V}$ is the set of mesh vertices, t is global translation, and R is global rotation (see Sec. 4). The objective function comprises of the following terms:

$$\begin{aligned} E(\Theta) = E_{data}(\Theta) + \lambda_{top} E_{top}(\Theta) + \lambda_{iso} E_{iso}(\Theta) \\ + \lambda_{geo} E_{geo}(\Theta) + \lambda_{reg} E_{reg}(\Theta), \end{aligned} \tag{10}$$

where $\lambda_{top}, \lambda_{iso}, \lambda_{geo}$ and λ_{reg} are scalar weights. The data term is further split into an event term and a silhouette term:

$$E_{data}(\Theta) = E_{event}(\Theta) + \lambda_{sil} E_{sil}(\Theta), \tag{11}$$

where λ_{sil} is another hyperparameter that balances the importance of each term. The event and regularisation terms remain the same as in (7) and (9), respectively. We now discuss the new terms of the objective function in detail.

5.3.1 Silhouette Term

The silhouette term matches the relevant events with the 2D projection of the nearest visible mesh vertices. The nearest vertices are calculated based on the shortest Euclidean distance between an event and the vertices in the 2D image:

$$E_{sil}(\Theta) = \sum_{e_b \in \hat{\mathcal{E}}} ||\pi(v_b) - (x_b, y_b)||_2^2, \tag{12}$$

where v_b is the 3D mesh vertex closest to the event e_b in the 2D space, (x_b, y_b) represents the 2D location of e_b, π denotes the 3D-to-2D projection and $\hat{\mathcal{E}}$ stands for the set of relevant events in the current event frame. The noisy events are filtered out by taking a 5×5 neighbourhood around each event in the event frame and then removing them if the number of events present in this region is below some threshold.

5.3.2 Topology-Preserving Term

The topology-preserving term demands that the relative positions of the current mesh vertices with respect to their neighbours remain the same as those of the template mesh, *i.e.,* it ensures spatial smoothness of the surface:

$$E_{top}(\Theta) = \sum_{\mathbf{i}=1}^{N} \sum_{\mathbf{j}\in\mathcal{N}_i} ||(v_i - v_j) - (\mathbf{v}_i - \mathbf{v}_j)||_2^2, \tag{13}$$

where N represents the total number of mesh vertices, v_i denotes the i-th vertex of the current mesh, $\mathbf{v}_i$ is the i-th vertex of the template mesh and $j \in \mathcal{N}_i$ indexes the points in the neighbourhood of the i-th vertex.

5.3.3 Isometric and Geodesic Terms

We consider isometric shape deformations, *i.e.*, that the area of the mesh or the distance between any two points on the surface of the mesh remain unchanged. Thus, the isometric term ensures that the mesh edge length between neighbouring vertices is preserved with respect to the template. The geodesic term, similarly, ensures that the distance between any two non-neighbouring vertices on the mesh surface remains preserved with respect to the template. Since computing this distance for all possible combinations of vertices can be computationally very expensive, we, therefore, uniformly sample vertex points on the template mesh surface in a coarse manner (we take every tenth point) and compute the geodesic loss for this grid of sparse points only:

$$E_{iso}(\Theta) = \sum_{\mathbf{i}=1}^{N} \sum_{\mathbf{j}\in\mathcal{N}_i} \big(||v_i - v_j|| - ||\mathbf{v}_i - \mathbf{v}_j||\big)^2, \tag{14}$$

$$E_{geo}(\Theta) = \sum_{\mathbf{i}\in\Omega} \sum_{\substack{\mathbf{j}\in\Omega,\\ j\neq i}} \big(d(v_i, v_j) - d(\mathbf{v}_i, \mathbf{v}_j)\big)^2, \tag{15}$$

where Ω denotes the grid of sub-sampled vertices and d denotes the geodesic distance between two points.

5.4. Initialisation

Depending on whether we have a generic mesh or a morphable model, we either initialise with the template mesh or the mean shape of the model. For the initial frame, we start from a known position. Regarding subsequent event frames, we initialise with the estimated parameters of the previous frame and optimise for all the parameters. For each frame, we first perform rigid fitting, *i.e.*, optimising for t and R only keeping the rest of the parameters fixed, and thereafter we optimise for the other parameters.

5.5. Implementation Details

We implement our method in *PyTorch* [15]. The optimiser used is *Adam* [11] with step size of $5 \cdot 10^{-4}$. The method takes $\approx$1.5 minutes per frame on NVIDIA V100 GPU. We balance the various energy terms with $\lambda = 10$, $\lambda_1 = 0.1$, $\lambda_{sil} = 0.01$, $\lambda_{top} = 1$, $\lambda_{iso} = 1$, $\lambda_{geo} = 1$, and $\lambda_{reg} = 10$.

6. Experimental Results

In this section, we report our experiments with the proposed non-rigid tracking methods using our differentiable event stream simulator. We consider three types of non-rigid objects in our experiments: a hand mesh, a deformable sheet (paper) and a spherical watertight mesh (ball). For hands, we use a 3D morphable hand model [21]. We compare to several existing state-of-the-art methods for RGB-based non-rigid tracking, including template-based methods of Tien Ngo *et al.* [13] and Direct-Dense-Deformable (DDD) [30], learning-based method Isometric Monocular GAN (IsMo-GAN) [25] and HandGraphCNN of Ge *et al.* [7] for 3D hand shape reconstruction. To ensure accurate initialisation for template-based methods (*i.e.,* in the first frames), we perform manual adjustment, if required.

6.1. Evaluation Metrics

For hands, we use the average 3D joint error as our metric for quantitative evaluation:

$$e_{joint3D} = \frac{1}{N} \sum_{i=1}^{N} \frac{\left\| J_{GT}^i - J_{rec}^i \right\|_F}{\left\| J_{GT}^i \right\|_F}, \tag{16}$$

where N is the number of frames and $\|\cdot\|_F$ denotes Frobenius norm; J_{GT} and J_{rec} denote the ground-truth and reconstructed 3D joint locations, respectively. For meshes, we employ average e_{3D} which differs from $e_{joint3D}$ in

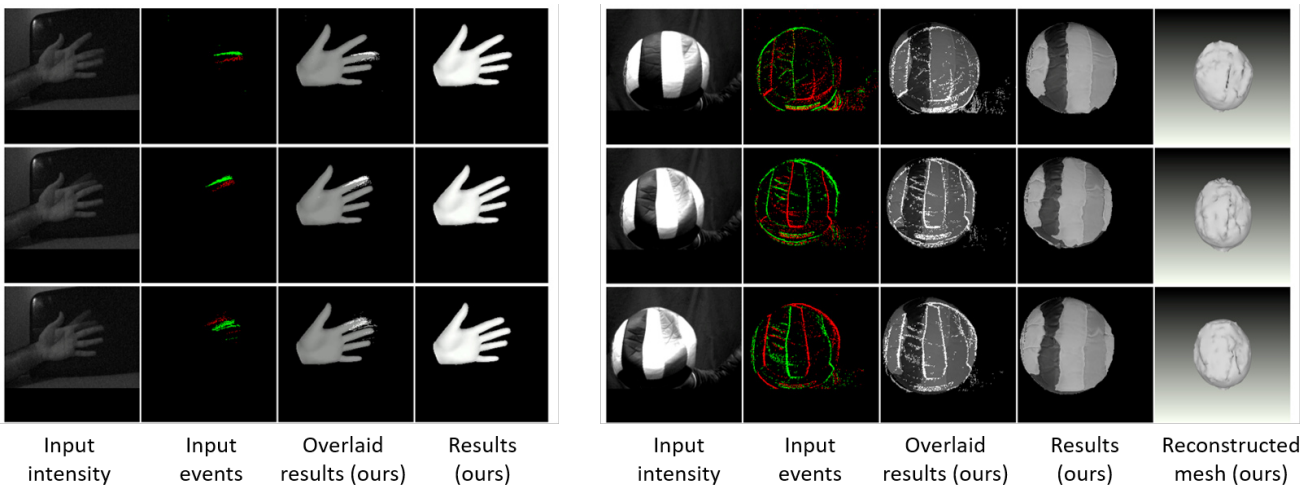

Figure 2. Results produced by our technique on the real hand (left) and real ball (right) sequences.

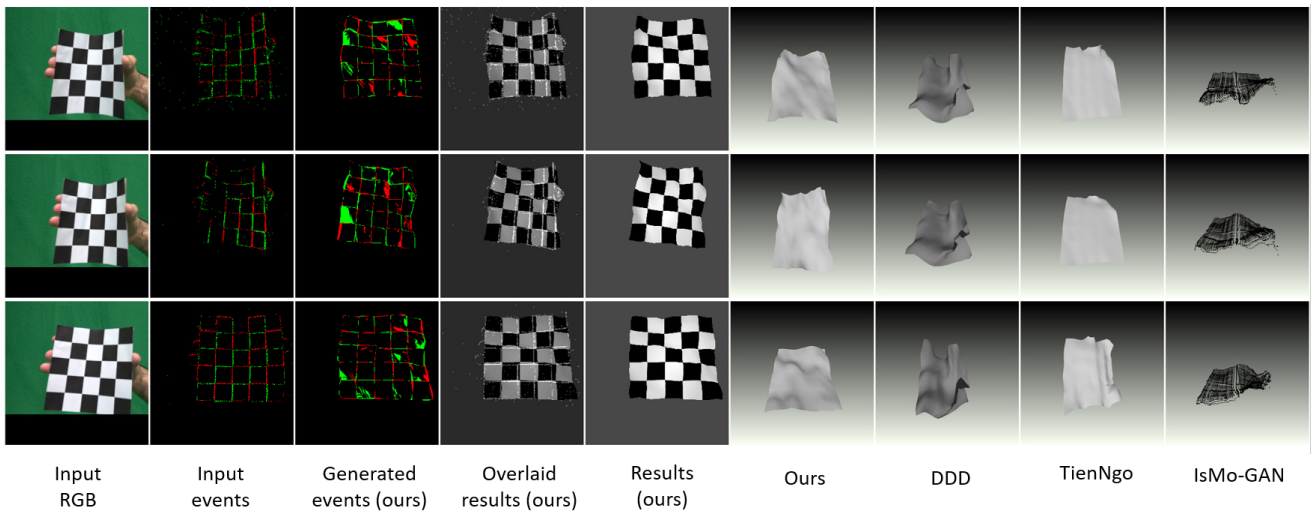

Figure 3. Results on the *real paper*. Ours produces more feasible reconstructions and fewer surface artefacts in the comparisons.

that it compares the dense reconstructed meshes S_{rec} with ground-truth meshes S_{GT} instead of sparse sets of joints. Both $e_{joint3D}$ and e_{3D} are reported after Procrustes alignment of the shapes, *i.e.,* with the resolved translational and rotational components.

6.2. Real Sequences

We record real sequences of the three classes of 3D objects: hands, paper (deformable sheet) and ball. Our setup comprises of DAVIS 240C for recording events and an additional RGB sensor (Sony RX0) that records coloured intensity images at 50 fps. The coloured intensity images are used for the methods being compared against our approach, since they are RGB image-based methods. We calibrate the cameras together and compute the extrinsics. For synchronisation, we use a flash. For a 36-seconds-long recording of a scene with significant motion, the events take up around 23 MB of storage space as compared to the corresponding video which takes up around 110 MB of space. We record a 20-seconds-long video for fast hand motion comprising of around 1500 event frames; another short 10-seconds video demonstrating the deformation of a volleyball, comprising of around 400 event frames and another 50-seconds-long video of a deforming sheet of paper with printed texture, comprising of 1100 event frames. The spatial resolution of event as well as intensity frames is set to 240×240 pixels. We use $T = 800$ events for hand and $T = 2000$ events for paper and ball, see (2), and set $C = e^{0.5}$ for real data on the $[0, 255]$ scale. Figs. 2–3 show results of various methods on the real sequences. Our technique produces realistic results that overlap well with the input event stream, whereas results of competing methods evince various surface artefacts and folding which are not observed in the input images.

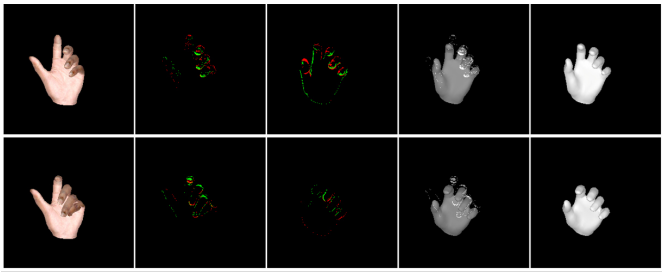

Figure 4. Results of our technique on the *synthetic hand* sequence.

6.3. Controlled Experiments

We prepare synthetic data with 3D ground truth for quantitative analysis. We render synthetic RGB image sequences and event stream data for each object. In the case of hands, we sample the pose parameter space using adaptive sampling similarly to Rebecq *et al.* [19], see our supplement for the details. and render the corresponding intensity images. For the generic meshes, we deform the meshes in Blender [4] and render the intensity images at 50 fps. Then, we generate synthetic event data from two consecutive images by subtracting them, followed by thresholding the difference image. The sequences contain 306 (*hand*), 250 (*paper*) and 200 (*ball*) frames. We use the following values for the time window T in (2): 400 events for *hand* and 1200 events for *paper* and *ball*, and set $C = 10$ for synthetic data.

For the *synthetic hand* sequence, we report $e_{joint3D}$ in Table 1. Our method achieves approximately twice as lower error as HandGraphCNN [7]. The latter is a learning-based approach operating on RGB images, on each frame individually, that does not assume accurate shape initialisation for one of the frames. Considering that the images, on which HandGraphCNN has been trained, are different from our rendered images which leads to domain gap, it is conceivable that HandGraphCNN does not outperform our method. Note that we cannot retrain HandGraphCNN on our data, because we render only a few hundreds of images (our method is unsupervised). Nevertheless, the strength of our approach comes from its use of the data term which is capable of tracking such highly articulated objects as human hands. The visual performance of our technique on the *synthetic hand* sequence is shown in Fig. 4.

For the *synthetic paper* sequence, the 3D reconstruction error is reported in Table 2. Our method is ranked first and we slightly outperform the method of Tien Ngo *et al.* [13] in e_{3D}. Both DDD and IsMo-GAN are not able to track the observed surfaces accurately, though DDD shows a slightly worse accuracy compared to [13]. Note that both Tien Ngo *et al.* [13] and DDD [30] do not suffer from domain gap of learning-based methods. On the other hand, IsMo-GAN is positioned as a generalisable approach, due to the powerful silhouette term, which allows it to reconstruct surfaces that are significantly different from those observed in the training set. Fig. 5 shows visualisations of the results obtained by the tested methods on the *synthetic paper* sequence.

For the *synthetic ball* sequence, we report e_{3D} in Table 3. We compare to DDD [30] which supports watertight meshes. On average, we outperform DDD by $\approx 30\%$ and our reconstructions reflect the deformations in the occluded parts more faithfully. The qualitative improvement in the results of our approach over DDD is shown in Fig. 6.

Method	$e_{joint3D}$	std. deviation
HandGraphCNN [7]	0.191	±0.055
Ours	**0.074**	±0.027

Table 1. Quantitative results on the *synthetic hand* sequence.

Method	e_{3D}	std. deviation
DDD [30]	0.266	±0.12
Tien Ngo *et al.* [13]	0.235	±0.158
IsMo-GAN [25]	0.384	±0.092
Ours	**0.232**	±0.135

Table 2. Quantitative results on the *synthetic paper* sequence.

Method	e_{3D}	std. deviation
DDD [30]	0.656	±0.151
Ours	**0.47**	±0.31

Table 3. Quantitative results on the *synthetic ball* sequence.

6.4. Ablation Study

We perform an ablation study for the proposed energy-based method for non-rigid 3D tracking. The summary of the attained e_{3D} is provided in Table 4. Here, we evaluate on the same synthetic sequences used in Sec. 6.3. We observe that using all terms produces the best results. Leaving out any of the energy terms leads to an accuracy drop. For the *ball* sequence, leaving out the silhouette term or both the isometric and topology preserving terms leads to the most significant accuracy drop. For the *hand* sequence, leaving out the no-event term leads to a significant accuracy drop. Please note that processing each of these sequences requires different energy terms (see Sec. 5.2 and Sec. 5.3).

7. Conclusion

We introduced the first differentiable event stream simulator which enables non-rigid 3D tracking of deformable objects. In contrast to previous event-based tracking techniques, our tracking approach does not require large training datasets or computation of explicit event trajectories.

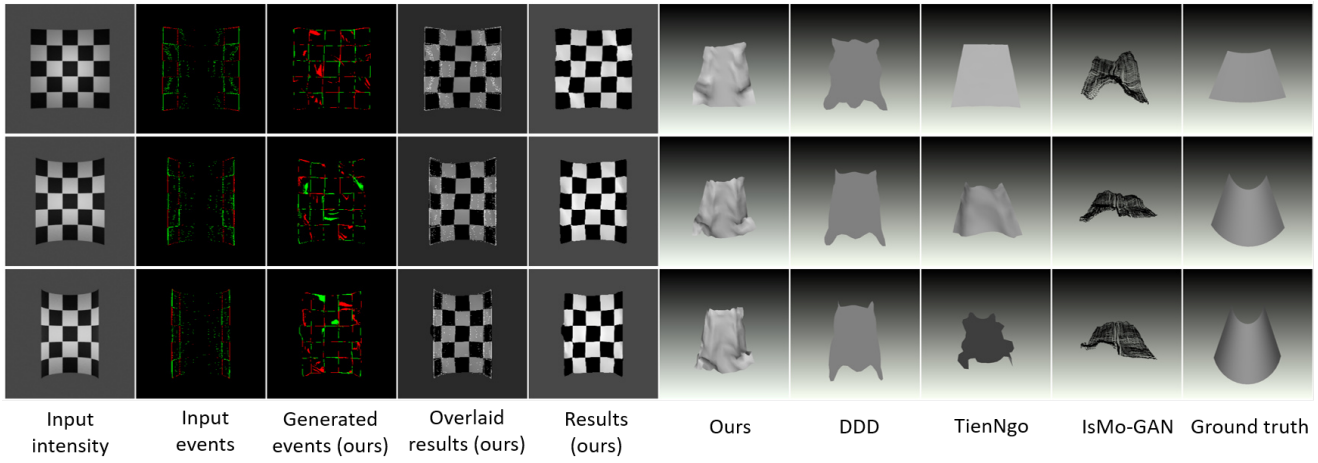

Figure 5. Results on the *synthetic paper* sequence. Our technique outperforms existing methods quantitatively (see Table 2).

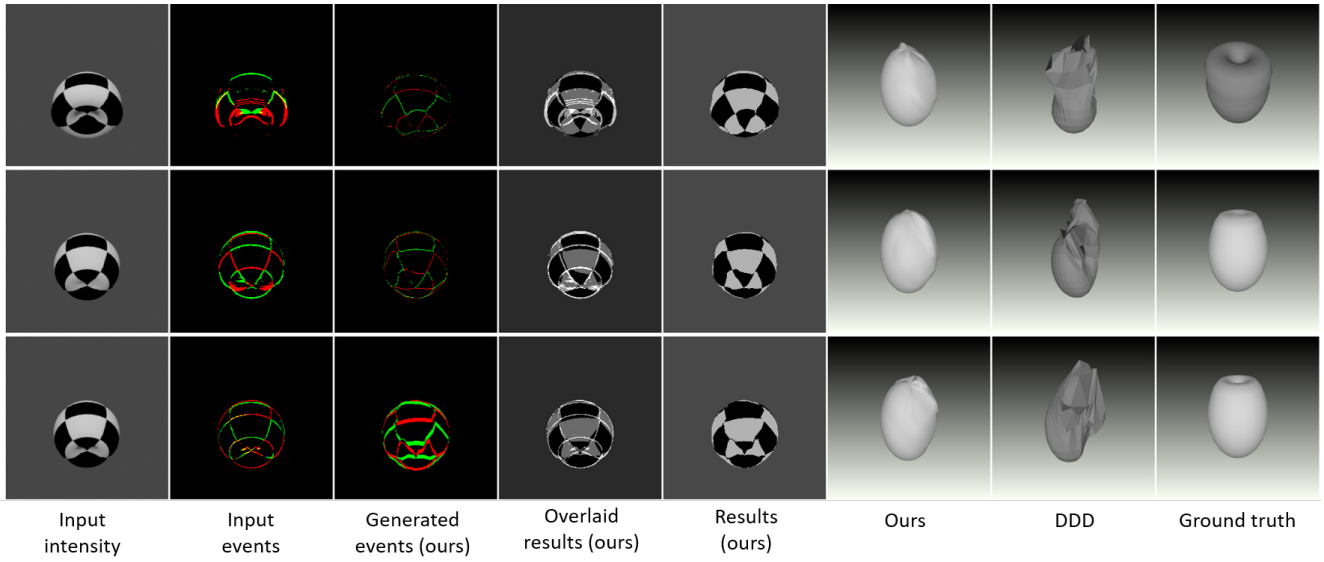

Figure 6. Results on the *synthetic ball* sequence. Our technique outperforms existing methods quantitatively (see Table 3).

	Method	e_{3D}	std. deviation
Ball	All terms	**0.467**	±0.312
	w/o Silhouette term	0.571	±0.326
	w/o Topological term	0.564	±0.307
	w/o Isometric terms	0.546	±0.303
	w/o Top + Iso terms	0.579	±0.293
Hand	All terms	**0.074**	±0.027
	w/o No-Event term	0.141	±0.055

Table 4. Ablative analysis on the *synthetic ball* and *hand*.

Instead, we leverage the differentiability of our event simulator in an analysis-by-synthesis optimisation framework. We demonstrated the generality of our method by performing quantitative and qualitative evaluation of our method for different deformable objects. Future work can address improving the processing speed and making it real-time.

One important conclusion which we draw from the experiments is that events not only pose additional challenges and add complexity for non-rigid 3D tracking, but they also can improve the tracking accuracy compared to the RGB-based methods. This is because events represent a more abstract data modality compared to RGB images, which we leverage with appropriate novel data terms and regularisers. We believe the proposed framework can be used for other tasks in event-based vision such as 2D tracking and gesture recognition. It can also be used as a differentiable component in supervised learning methods. We hope our framework inspires more future work in event-based vision.

Acknowledgement. This work was supported by the ERC Consolidator Grant 770784.

References

[1] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition (CVPR)*, 2000.

[2] Samuel Bryner, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. Event-based, direct camera tracking from a photometric 3D map using nonlinear optimization. In *International Conference on Robotics and Automation (ICRA)*, 2019.

[3] Enrico Calabrese, Gemma Taverni, Christopher Awai Easthope, Sophie Skriabine, Federico Corradi, Luca Longinotti, Kynan Eng, and Tobi Delbruck. Dhp19: Dynamic vision sensor 3d human pose dataset. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[4] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, 2018.

[5] Katerina Fragkiadaki, Marta Salas, Pablo Arbelaez, and Jitendra Malik. Grouping-based low-rank trajectory completion and 3d reconstruction. In *Neural Information Processing Systems (NeurIPS)*, 2014.

[6] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jorg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[7] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Vladislav Golyanik, André Jonas, and Didier Stricker. Consolidating segmentwise non-rigid structure from motion. In *Machine Vision Applications (MVA)*, 2019.

[9] Nail Gumerov, Ali Zandifar, Ramani Duraiswami, and Larry S. Davis. Structure of applicable surfaces from single views. In *European Conference on Computer Vision (ECCV)*, pages 482–496, 2004.

[10] Hanme Kim, Stefan Leutenegger, and Andrew J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European Conference on Computer Vision (ECCV)*, 2016.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv e-prints*, 2014.

[12] Suryansh Kumar, Anoop Cherian, Yuchao Dai, and Hongdong Li. Scalable dense non-rigid structure-from-motion: A grassmannian perspective. In *Computer Vision and Pattern Recognition (CVPR)*, pages 254–263, 2018.

[13] Dat Tien Ngo, Sanghyuk Park, Anne Jorstad, Alberto Crivellaro, Chang D. Yoo, and Pascal Fua. Dense image registration and deformable surface reconstruction in presence of occlusions and minimal texture. In *International Conference on Computer Vision (ICCV)*, 2015.

[14] Jonas Östlund, Aydin Varol, Dat Tien Ngo, and Pascal Fua. Laplacian meshes for monocular 3d shape recovery. In *European Conference on Computer Vision (ECCV)*, 2012.

[15] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.

[16] Mathieu Perriollat, Richard Hartley, and Adrien Bartoli. Monocular template-based reconstruction of inextensible surfaces. *International Journal of Computer Vision (IJCV)*, 95(2):124–137, 2011.

[17] Neng Qian, Jiayi Wang, Franziska Mueller, Florian Bernard, Vladislav Golyanik, and Christian Theobalt. Html: A parametric hand texture model for 3d hand reconstruction and personalization. In *European Conference on Computer Vision (ECCV)*, 2020.

[18] Henri Rebecq, Guillermo Gallego, Elias Mueggler, and Davide Scaramuzza. EMVS: Event-based multi-view stereo—3D reconstruction with an event camera in real-time. *International Journal of Computer Vision (IJCV)*, 126:1394–1414, 2018.

[19] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: an open event camera simulator. *Conference on Robot Learning (CoRL)*, 2018.

[20] Christian Reinbacher, Gottfried Munda, and Thomas Pock. Real-time panoramic tracking for event cameras. In *International Conference on Computational Photography (ICCP)*, 2017.

[21] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *SIGGRAPH Asia*, 36(6), 2017.

[22] Viktor Rudnev, Vladislav Golyanik, Jiayi Wang, Hans-Peter Seidel, Franziska Mueller, Mohamed Elgharib, and Christian Theobalt. Eventhands: Real-time neural 3d hand reconstruction from an event stream. *arXiv e-prints*, 2020.

[23] Mathieu Salzmann, Julien Pilet, Slobodan Ilic, and Pascal Fua. Surface deformation models for nonrigid 3d shape recovery. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(8):1481–1487, 2007.

[24] Stephan Schraml, Ahmed Nabil Belbachier, and Horst Bischof. Event-driven stereo matching for real-time 3d panoramic vision. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[25] Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Didier Stricker. IsMo-GAN: Adversarial learning for monocular non-rigid 3d reconstruction. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[26] Vikramjit Sidhu, Edgar Tretschk, Vladislav Golyanik, Antonio Agudo, and Christian Theobalt. Neural dense non-rigid structure from motion with latent space constraints. In *European Conference on Computer Vision (ECCV)*, 2020.

[27] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(5):878–892, 2008.

[28] Aggeliki Tsoli and Antonis A. Argyros. Patch-based reconstruction of a textureless deformable 3d surface from a single

rgb image. In *International Conference on Computer Vision Workshops (ICCVW)*, 2019.

[29] Lan Xu, Weipeng Xu, Vladislav Golyanik, Marc Habermann, Lu Fang, and Christian Theobalt. Eventcap: Monocular 3d capture of high-speed human motions using an event camera. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[30] Rui Yu, Chris Russell, Neill D. F. Campbell, and Lourdes Agapito. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. In *International Conference on Computer Vision (ICCV)*, 2015.