This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Real-Time 6DOF Pose Relocalization for Event Cameras with Stacked Spatial LSTM Networks

Anh Nguyen
Istituto Italiano di Tecnologia, Italy
Anh.Nguyen@iit.it

Thanh-Toan Do
University of Liverpool and AIOZ Pte Ltd
thanh-toan.do@liverpool.ac.uk

Darwin G. Caldwell
Istituto Italiano di Tecnologia, Italy
Darwin.Caldwell@iit.it

Nikos G. Tsagarakis
Istituto Italiano di Tecnologia, Italy
Nikos.Tsagarakis@iit.it

Abstract

We present a new method to relocalize the 6DOF pose of an event camera solely based on the event stream. Our method first creates the event image from a list of events that occurs in a very short time interval, then a Stacked Spatial LSTM Network (SP-LSTM) is used to learn the camera pose. Our SP-LSTM is composed of a CNN to learn deep features from the event images and a stack of LSTM to learn spatial dependencies in the image feature space. We show that the spatial dependency plays an important role in the relocalization task with event images and the SP-LSTM can effectively learn this information. The extensively experimental results on a publicly available dataset show that our approach outperforms recent state-of-the-art methods by a substantial margin, as well as generalizes well in challenging training/testing splits. The source code and trained models are available at https://github.com/nqanh/pose_relocalization.

1. INTRODUCTION

Inspired by human vision, the event-based cameras asynchronously capture an event whenever there is a brightness change in a scene [2]. An event is simply composed of a pixel coordinate, its binary polarity value, and the timestamp when the event occurs. This differs from the frame-based cameras where an entire image is acquired at a fixed time interval. Based on its novel design concept, the event-based camera can rapidly stream events (i.e., at microsecond speeds). This is superior to frame-based cameras which usually sample images at millisecond rates [3]. This novel ability makes the event cameras more suitable for the high-speed robotic applications that require low latency and high dynamic range from the visual data.

Although the event camera creates a paradigm shift in solving real-time visual problems, its data come extremely quickly without the intensity information usually found in an image. Each event also carries very little information (i.e., the pixel coordinate, the polarity value and the timestamp) when it occurs. Therefore, it is not trivial to apply standard computer vision techniques to event data. Recently, the event camera is gradually becoming more popular in the computer vision and robotics community. Many problems such as camera calibration and visualization [17], 3D reconstruction [14], simultaneous localization and mapping (SLAM) [28], and pose tracking [16] have been actively investigated.

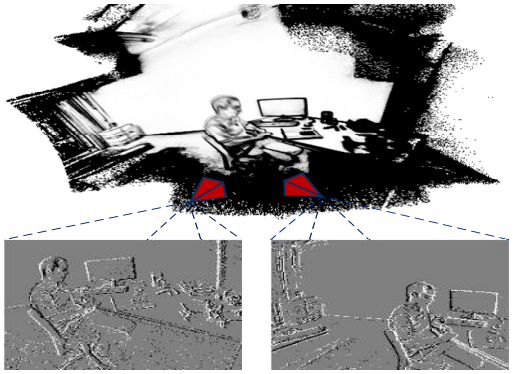


Figure 1. Pose relocalization for event cameras. We propose to create event images from lists of events and relocalize the 6DOF camera poses from these images using a deep neural network.

Our goal in this work is to develop a new method, which relocalizes the 6 Degrees of Freedom (6DOF) pose of the event camera using a deep learning approach. The problem of effectively and accurately interpreting the pose of the camera plays an important role in many robotic applications such as navigation and manipulation. However, in practice it is challenging to estimate the pose of the event

camera since it can capture a lot of events in a short time interval, yet each event does not have enough information to perform the estimation. We propose to form a list of events into an event image and regress the camera pose from this image with a deep neural network. The proposed approach can accurately recover the camera pose directly from the input events, without the need for additional information such as the 3D map of the scene or inertial measurement data.

In computer vision, Kendall et al. [12] introduced a first deep learning framework to retrieve the 6DOF camera pose from a single image. The authors in [12] showed that compared to the traditional keypoint approaches, using CNN to learn deep features resulted in a system that is more robust in challenging scenarios such as noisy or uncleared images. Recently, the work in [11] introduced a method that used a geometry loss function to learn the spatial dependencies. In this paper, we employ the same concept, using CNN to learn deep features, however unlike [11] that builds a geometry loss function based on the 3D points in the scene, we use an SP-LSTM network to encode the geometry information. Since the event images are usually noisy and do not have the useful visual information such as color, density, etc. as in normal images, encoding the geometric structure and spatial features from the data is the key step in the relocalization task.

Next, we review the related work in Section 2, followed by a description of the event data and event images in Section 3. The SP-LSTM network is introduced in Section 4. In Section 5, we present the extensive experimental results. Finally, we conclude the paper in Section 6.

2. Related Work

The event camera is particularly suitable for real-time motion analysis or high-speed robotic applications since it has low latency [17]. Early work on event cameras used this property to track an object to provide fast visual feedback to control a simple robotic system [6]. The authors in [16] set up an onboard perception system with an event camera for 6DOF pose tracking of a quadrotor. Using the event camera, the quadrotor's poses can be estimated with respect to a known pattern during high-speed maneuvers. Recently, a 3D SLAM system was introduced in [28] by fusing frame-based RGB-D sensor data with event data to produce a sparse stream of 3D points.

In [13], the authors presented a method to estimate the rotational motion of the event camera using two probabilistic filters. Recently, Kim et al. [14] extended this system with three filters that simultaneously estimate the 6DOF pose of the event camera, the depth, and the brightness of the scene. The work in [8] introduced a method to directly estimate the angular velocity of the event camera based on a contrast maximization design without requiring optical flow or image intensity estimation. Reinbacher et al. [22] introduced a method to track an event camera based on a panoramic setting that only relies on the geometric properties of the event stream. More recently, the authors in [30] [21] proposed to fused events with IMU data to accurately track the 6DOF camera pose.

In computer vision, 6DOF camera pose relocalization is a well-known problem. Recent research trends investigate the capability of deep learning for this problem [5, 7, 11, 12, 15, 18, 25–27, 29]. Kendall et al. [12] introduced a first deep learning framework to regress the 6DOF camera pose from a single input image. The work of [10] used Bayesian uncertainty to correct the camera pose. Recently, the authors in [11] introduced a geometry loss function based on 3D points from a scene, to let the network encode the geometry information during the training phase. LSTM is also widely used in many tasks such as object captioning [19], pose relocalization [26], and video translating [20]. The main advantage of the deep learning approach is that the deep network can effectively encode the features from the input images, without relying on the hand-designed features.

In this work, we first create an event image from a list of events. A deep network composed of a CNN and an SP-LSTM is then trained end-to-end to regress the camera pose. While most of recent work [15, 18, 25, 29] focus on using CNN to regress the 6DOF pose, we combine both CNN and LSTM to encode the geometric structure of the event images. Our approach is more similar to [11, 26], however unlike [11] that used only CNN with a geometry loss function that required the 3D points from the scene, or [26] that used four parallel LSTM to encode the geometry information, we use SP-LSTM to learn spatial dependencies from event images. We show that encoding the geometric structure with SP-LSTM is the key step that leads to the improvement in the task of 6DOF relocalization with event images.

3. Event Data

3.1. Event Camera

Instead of capturing an entire image at a fixed time interval as in standard frame-based cameras, the event cameras only capture a single event at a timestamp based on the brightness changes at a local pixel. In particular, an event e is a tuple $e =< e_t, (e_x, e_y), e_\rho >$ where e_t is the timestamp of the event, (e_x, e_y) is the pixel coordinate and $e_\rho = \pm 1$ is the polarity that denotes the brightness change at the current pixel. The events are transmitted asynchronously with their timestamps using a sophisticated digital circuitry. Recent event cameras such as DAVIS 240C [2] also provide IMU data and global-shutter images. In this work, we only use the event stream as the input for our deep network.

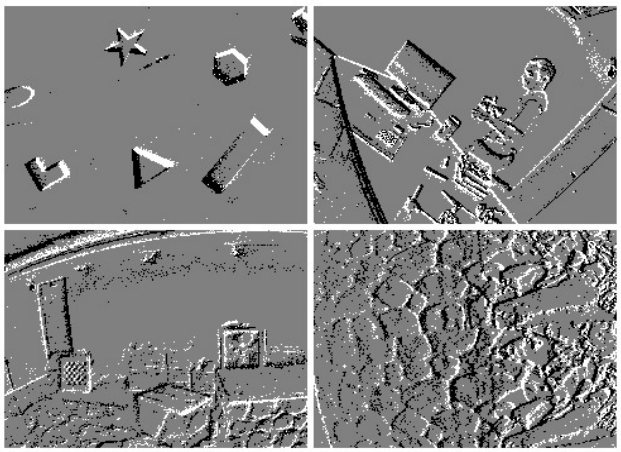


Figure 2. Examples of event images. Since the events mainly occur around the edge of the objects, the event images are clearer on simple scenes (top row), while more disorder on cluttered scenes (bottom row).

3.2. From Events to Event Images

Since a single event only contains a binary polarity value of a pixel and its timestamp, it does not carry enough information to estimate the 6DOF pose of the camera. In order to make the pose relocalization problem using only the event data becomes feasible, similar to [8] we assume that n events in a very short time interval will have the same camera pose. This assumption is based on the fact that the event camera can capture many events in a short period, while in that very short time interval, the poses of the camera can be considered as unchanging significantly. From a list of n events, we reconstruct an event image $I \in \mathbb{R}^{h \times w}$ (where h and w are the height and width resolution of the event camera) based on the value of the polarity e_ρ as follows:

$$I(e_x, e_y) = \begin{cases} 0, & \text{if } e_\rho = -1 \\ 1, & \text{if } e_\rho = 1 \\ 0.5, & \text{otherwise} \end{cases} \tag{1}$$

This conversion allows us to transform a list of events to an image and apply traditional computer vision techniques to event data. Since the events mainly occur around the edge of the scene, the event images are clearer on simple scenes, while more disorder on cluttered scenes. Fig. 2 shows some examples of event images. In practice, the parameter n plays an important role since it affects the quality of the event images, which are used to train and infer the camera pose. We analyze the effect of this parameter to the pose relocalization results in Section 5-D.

4. Pose Relocalization for Event Camera

4.1. Problem Formulation

Inspired by [12] [10], we solve the 6DOF pose relocalization task as a regression problem using a deep neural network. Our network is trained to regress a pose vector $\mathbf{y} = [\mathbf{p}, \mathbf{q}]$ with $\mathbf{p}$ represents the camera position and $\mathbf{q}$ represents the orientation in 3D space. We choose quaternion to represent the orientation since we can easily normalize its four dimensional values to unit length to become a valid quaternion. In practice, the pose vector $\mathbf{y}$ is seven dimensional and is defined relatively to an arbitrary global reference frame. The groundtruth pose labels are obtained through an external camera system [17] or structure from motion [12].

4.2. Stacked Spacial LSTM

We first briefly describe the Long-Short Term Memory (LSTM) network [9], then introduce the Stacked Spatial LSTM and the architecture to estimate the 6DOF pose of event cameras. The core of the LSTM is a memory cell which has the gate mechanism to encode the knowledge of previous inputs at every time step. In particular, the LSTM takes an input $\mathbf{x}_t$ at each time step t, and computes the hidden state $\mathbf{h}_t$ and the memory cell state $\mathbf{c}_t$ as follows:

$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \\ \mathbf{g}_t &= \phi(\mathbf{W}_{xg}\mathbf{x}_t + \mathbf{W}_{hg}\mathbf{h}_{t-1} + \mathbf{b}_g) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\ \mathbf{h}_t &= \mathbf{o}_t \odot \phi(\mathbf{c}_t) \end{aligned} \tag{2}$$

where $\odot$ represents element-wise multiplication; the function σ is the sigmoid non-linearity, and ϕ is the hyperbolic tangent non-linearity. The weight $\mathbf{W}$ and bias $\mathbf{b}$ are trained parameters. With this gate mechanism, the LSTM network can choose to remember or forget information for long periods of time, while is still robust against vanishing or exploding gradient problems.

Although the LSTM network is widely used to model temporal sequences, in this work we use the LSTM network to learn spatial dependencies in image feature space. The spatial LSTM has the same architecture as normal LSTM, however, unlike normal LSTM where the input is from the time axis of the data (e.g., a sequence of words in a sentence or a sequence of frames in a video), the input of spatial LSTM is from feature vectors of the image. Recent work showed that the spatial LSTM can further improve the results in many tasks such as music classification [4] or image modeling [24]. Stacked Spatial LSTM is simply a stack of several LSTM layers, in which each layer aims at learning the spatial information from image features. The intuition is

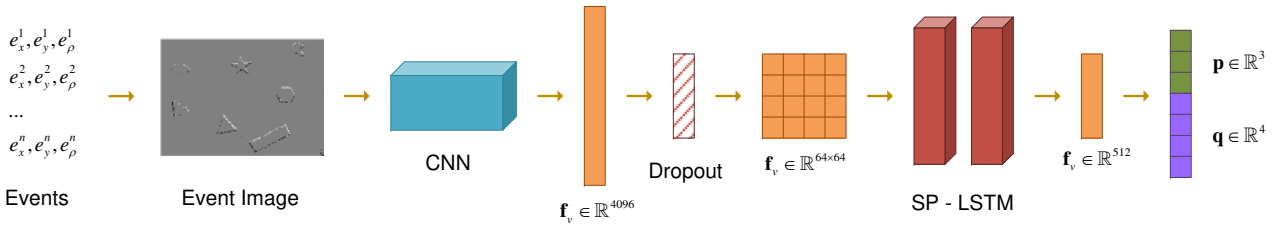


Figure 3. An overview of our 6DOF pose relocalization method for event cameras. We first create an event image from a list of events, then a CNN is used to learn deep features from this image. The image feature vector is reshaped and fed to a SP-LSTM network with 256 hidden units. Finally, the output of SP-LSTM is fed to a fully connected layer with 512 neurons, following by another fully connected layer with 7 neurons to regress the pose vector.

that higher LSTM layers can capture more abstract concepts in the image feature space, hence improving the results.

4.3. Pose Relocalization with Stacked Spatial LSTM

Our pose regression network is composed of two components: a deep CNN and an SP-LSTM network. The CNN network is used to learn deep features from the input event images. After the last layer of the CNN network, we add a dropout layer to avoid overfitting. The output of this CNN network is reshaped and fed to the SP-LSTM module. A fully connected layer is then used to discard the relationships in the output of LSTM. Here, we note that we only want to learn the spatial dependencies in the image features through the input of LSTM, while the relationships in the output of LSTM should be discarded since the components in the pose vector are independent. Finally, a linear regression layer is appended at the end to regress the seven dimensional pose vector. Fig. 3 shows an overview of our approach.

In practice, we choose the VGG16 [23] network as our CNN. We first discard its last softmax layer and add a dropout layer with the rate of 0.5 to avoid overfitting. The event image features are stored in the last fully connected layer in a 4096 dimensional vector. We reshape this vector to 64×64 in order to feed to the LSTM module with 256 hidden units. Here, we can consider that the inputs of LSTM are from 64 "feature sentences", each has 64 "words", and the spatial dependencies are learned from these sentences. We then add another LSTM network to create an SP-LSTM with 2 layers. The output of SP-LSTM module is fed to a fully connected layer with 512 neurons, following by another fully connected layer with 7 neurons to regress the pose vector. We choose the SP-LSTM network with 2 layers since it is a good balance between accuracy and training time.

4.4. Training

To train the network end-to-end, we use the following objective loss function:

$$\mathcal{L}(I) = \|\hat{\mathbf{p}} - \mathbf{p}\|_2 + \|\hat{\mathbf{q}} - \mathbf{q}\|_2 \quad (3)$$

where $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$ are the predicted position and orientation from the network. In [11], the authors proposed to use a geometry loss function to encode the spatial dependencies from the input. However, this approach required a careful initialization and needed a list of 3D points to measure the projection error of the estimated pose, which is not available in the groundtruth of the dataset we use in our experiment.

For simplicity, we choose to normalize the quaternion to unit length during testing phrase, and use Euclidean distance to measure the difference between two quaternions as in [12]. Theoretically, this distance should be measured in spherical space, however, in practice the deep network outputs the predicted quaternion $\hat{\mathbf{q}}$ close enough to the groundtruth quaternion $\mathbf{q}$, making the difference between the spherical and Euclidean distance insignificant. We train the network for 1400 epochs using stochastic gradient descent with 0.9 momentum and $1e-6$ weight decay. The learning rate is empirically set to $1e-5$ and kept unchanging during the training. It takes approximately 2 days to train the network from scratch on a Tesla P100 GPU.

5. EXPERIMENTS

5.1. Dataset

We use the event camera dataset that was recently introduced in [17] for our experiment. This dataset included a collection of scenes captured by a DAVIS camera in indoor and outdoor environments. The indoor scenes of this dataset have the groundtruth camera poses from a motion-capture system with sub-millimeter precision at 200 Hz. We use the timestamp of the motion-capture system to create event images. All the events with the timestamp between t and $t+1$ of the motion-capture system are grouped as one event image. Without the loss of generality, we consider the groundtruth pose of this event image is the camera pose that was taken by the motion-capture system at time $t+1$. This assumption technically limits the speed of the event camera to the speed of the motion-capture system (i.e. 200 Hz), however it allows us to use the groundtruth poses with sub-millimeter precision from the motion-capture system.

Random Split As the standard practice in the

Table 1. Pose Relocalization Results - Random Split

	PoseNet [12]	Bayesian PoseNet [10]	Pairwise-CNN [15]	LSTM-Pose [26]	SP-LSTM (ours)
shapes_rotation	$0.109m$, $7.388°$	$0.142m$, $9.557°$	$0.095m$, $6.332°$	$0.032m$, $4.439°$	$0.025m$, $2.256°$
box_translation	$0.193m$, $6.977°$	$0.190m$, $6.636°$	$0.178m$, $6.153°$	$0.083m$, $6.215°$	$0.036m$, $2.195°$
shapes_translation	$0.238m$, $6.001°$	$0.264m$, $6.235°$	$0.201m$, $5.146°$	$0.056m$, $5.018°$	$0.035m$, $2.117°$
dynamic_6dof	$0.297m$, $9.332°$	$0.296m$, $8.963°$	$0.245m$, $5.962°$	$0.097m$, $6.732°$	$0.031m$, $2.047°$
hdr_poster	$0.282m$, $8.513°$	$0.290m$, $8.710°$	$0.232m$, $7.234°$	$0.108m$, $6.186°$	$0.051m$, $3.354°$
poster_translation	$0.266m$, $6.516°$	$0.264m$, $5.459°$	$0.211m$, $6.439°$	$0.079m$, $5.734°$	$0.036m$, $2.074°$
Average	$0.231m$, $7.455°$	$0.241m$, $7.593°$	$0.194m$, $6.211°$	$0.076m$, $5.721°$	$0.036m$, $2.341°$

pose relocalization task [12], we *randomly* select 70% of the event images for training and the remaining 30% for testing. We use 6 sequences (shapes_rotation, box_translation, shapes_translation, dynamic_6dof, hdr_poster, poster_translation) for this experiment. These sequences are selected to cover different camera motions and scene properties.

Novel Split To demonstrate the generalization ability of our SP-LSTM network, we also conduct the experiment using the novel split. In particular, from the original event images sequence, we select *the first* 70% of the event images for training, then *the rest* 30% for testing. In this way, we have two independent sequences on the same scene (i.e., the training sequence is selected from timestamp t_0 to t_{70}, and the testing sequence is from timestamp t_{71} to t_{100}). We use three sequences from the shapes scene (shapes_rotation, shapes_translation, shapes_6dof) in this novel split experiment to compare the results when different camera motions are used.

We note that in both the random split and novel split strategies, after having the training/testing set, our SP-LSTM network selects the event image randomly for training/testing, and no sequential information between event images is needed. This is the key difference between our approach and the sequential methods that need two or more consecutive frames [5] [25]. Moreover, unlike the methods in [30] [21] that need the inertial measurement data, our SP-LSTM *only* uses the event images as the input.

5.2. Baseline

We compare our experimental results with the following recent state-of-the-art methods in computer vision: PoseNet [12], Bayesian PoseNet [10], Pairwise-CNN [15], and LSTM-Pose [26]. We reuse the source code provided by the authors of PoseNet, Bayesian PoseNet and Pairwise-CNN, while reimplementing LSTM-Pose since there is no public source code of this method. We note that all these methods use only the event images as the input and no further information such as 3D map of the environment or inertial measurements is needed.

For each sequence, we report the median error of the estimated poses in position and orientation separately. The predicted position is compared with the groundtruth using the Euclidean distance, while the predicted orientation is normalized to unit length before comparing with the groundtruth. The errors are measured in m and deg for the position and orientation, respectively.

5.3. Random Split Results

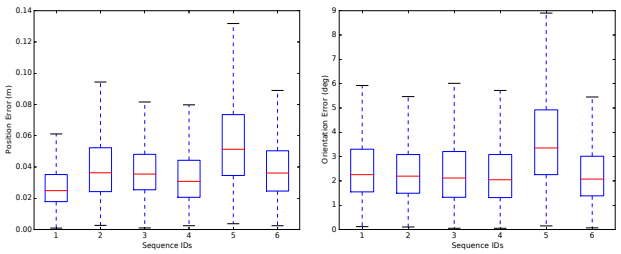
Table 1 summarizes the median error on 6 sequences using the random split strategy. From this table, we notice that the pose relocalization results are significantly improved using our SP-LSTM network in comparison with the baselines that used only CNN [12] [10] [15]. Our SP-LSTM achieves the lowest error in all sequences. In particular, SP-LSTM achieves ($0.036m$, $2.341°$) in median error on average of all sequences, while PoseNet, Bayesian PoseNet, Pairwise-CNN and LSTM-Pose results are ($0.231m$, $7.455°$), ($0.241m$, $7.593°$), ($0.194m$, $6.211°$), and ($0.076m$, $5.721°$), respectively. Overall, our proposed method improves around 2 times in both position error and orientation error, compared to the runner-up LSTM-Pose [26]. This demonstrates that the spatial dependencies play an important role in the camera pose relocalization process and our SP-LSTM successfully learns these dependencies, hence significantly improves the results. We also notice that Pairwise-CNN performs better than PoseNet and Bayesian PoseNet, while the uncertainty estimation in Bayesian PoseNet cannot improve the pose relocalization results for event images.

From Table 1, we notice that the pose relocalization results also depend on the properties of the scene in each sequence. Due to the design mechanism of the event-based camera, the events are mainly captured around the contours of the scene. In cluttered scenes, these contours are ambiguous due to non-meaningful texture edge information. Therefore, the event images created from events in these scenes are very noisy. As the results, we have observed that for sequences in cluttered or

Table 2. Pose Relocalization Results - Novel Split

	PoseNet [12]	Bayesian PoseNet [10]	Pairwise-CNN [15]	LSTM-Pose [26]	SP-LSTM (ours)
shapes_rotation	$0.201m$, $12.499°$	$0.164m$, $12.188°$	$0.187m$, $10.426°$	$0.061m$, $7.625°$	$0.045m$, $5.017°$
shapes_translation	$0.198m$, $6.969°$	$0.213m$, $7.441°$	$0.225m$, $11.627°$	$0.108m$, $8.468°$	$0.072m$, $4.496°$
shapes_6dof	$0.320m$, $13.733°$	$0.326m$, $13.296°$	$0.314m$, $13.245°$	$0.096m$, $8.973°$	$0.078m$, $5.524°$
Average	$0.240m$, $11.067°$	$0.234m$, $10.975°$	$0.242m$, $11.766°$	$0.088m$, $8.355°$	$0.065m$, $5.012°$

dense scenes (e.g. hdr_poster), the pose relocalization error is higher than sequences from the clear scenes (e.g. shapes_rotation, shapes_translation). We also notice that dynamic objects (e.g. as in dynamic_6dof scene) also affect the pose relocalization results. While PoseNet, Bayesian Posenet, and Pairwise-CNN are unable to handle the dynamic objects and have high position and orientation errors, our SP-LSTM gives reasonable results in this sequence. It demonstrates that by effectively learning the spatial dependencies with SP-LSTM, the results in such difficult cases can be improved.



(a) (b)

Figure 4. Error distribution of the pose relocalization results of our SP-LSTM network using random split. **(a)** Position error distribution. **(b)** Orientation error distribution. Sequences IDs are: 1-shapes_rotation, 2-box_translation, 3-shapes_translation, 4-dynamic_6dof, 5-hdr_poster, 6-poster_translation.

Error Distribution Fig. 4 shows the position and orientation error distributions of our SP-LSTM network. Each box plot represents the error for one sequence. We recall that the top and bottom of a box are the first and third quartiles that indicate the interquartile range (IQR). The band inside the box is the median. We notice that the IQR of position error of all sequences (except the hdr_poster) is around $0.02m$ to $0.05m$, while the maximum error is around $0.09m$. The IQR of orientation error is in the range $1.5°$ to $3.5°$, and the maximum orientation error is only $6°$. In all 6 sequences in our experiment, the hdr_poster gives the worst results. This is explainable since this scene is a dense scene, hence the event images have uncleared structure and very noisy. Therefore, it is more difficult for the network to learn and predict the camera pose from these images.

5.4. Novel Split Results

Table 2 summarizes the median and average error on 3 sequences using the novel split strategy. This table clearly shows that our SP-LSTM results outperform other state-of-the-art methods by a substantial margin. Our SP-LSTM achieves the lowest median error in both 3 sequences in this experiment, while the errors of PoseNet, Bayesian PoseNet and Pairwise-CNN remain high. In particular, the median error of our SP-LSTM is only ($0.065m$ and $5.012°$) in average, compared to ($0.240m$, $11.067°$), ($0.234m$, $10.975°$), and ($0.242m$, $11.766°$) from PoseNet, Bayesian PoseNet, and Pairwise-CNN, respectively. These results confirm that by learning the spatial relationship in the image feature space, the pose relocalization results can be significantly improved. Table 2 also shows that the dominating motion of the sequence also affects the results, for example, the translation error in the shapes_translation sequence is higher than shapes_rotation, and vice versa for the orientation error.

Compared to the pose relocalization errors using the random split (Table 1), the relocalization errors using the novel split are generally higher. This is explainable since the testing set from the novel split is much more challenging. We recall that in the novel split, the testing set is selected from the last 30% of the event images. This means we do not have the "neighborhood" relationship between the training and testing images. In the random split strategy, the testing images can be very close to the training images since we select the images randomly from the whole sequence for training/testing. This does not happen in the novel split strategy since the training and testing set are two separated sequences. Despite this challenging setup, our SP-LSTM still is able to regress the camera pose and achieves reasonable results. This shows that the network successfully encodes the geometry of the scene during training, hence generalizes well during the testing phase.

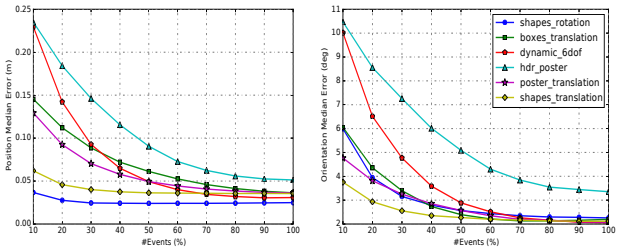
Features for Event Images In both random and novel split experiments, we have observed that LSTM-Pose [26] and our SP-LSTM consistently outperform other CNN-based methods. This shows that encoding spatial features from event images plays an important role in this task.

Compared to normal images, event images are noisy and do not have the useful visual information such as color, density, etc. Therefore, we can only rely on the geometric structure and spatial features of the image. Both LSTM-Pose [26] and our SP-LSTM are designed to focus on learning this information, hence achieving higher accuracy than CNN-based methods [12] [10] [15] which only rely on deep features alone. Overall, our SP-LSTM also outperforms LSTM-Pose since the use of stack of LSTM layers can further encode the geometric structure of the input images.

To conclude, the extensive experimental results from both the random split and novel split setup show that our SP-LSTM network successfully relocalizes the event camera pose using only the event image. The key reason that leads to the improvement is the use of stacked spatial LSTM to learn the spatial relationship in the image feature space. The experiments using the novel split setup also confirm that our SP-LSTM successfully encodes the geometry of the scene during the training and generalizes well during the testing. Furthermore, our SP-LSTM also has very fast inference time and requires only the event image as the input to relocalize the camera pose.

Reproducibility We implement the proposed method using Keras with Tensorflow framework [1]. The testing time for each new event image using our implementation is around $5ms$ on a Tesla P100 GPU, which is comparable to the real-time performance of PoseNet, while the Bayesian PoseNet takes longer time (approximately $240ms$) due to the uncertainty analysis process. To encourage further research, our source code and trained models are available at https://github.com/nqanh/pose_relocalization.

5.5. Robustness to Number of Events



(a) (b)

Figure 5. Robustness to number of events. **(a)** Position median error. **(b)** Orientation median error. The position and orientation errors of our SP-LSTM network do not significantly drop when we use more than 60% of all events to create the event images.

In this work, we assume that n events occurring between two timestamps of the external camera system will have the same camera pose. Although this assumption is necessary to use the groundtruth poses to train the network, it limits the speed of the event-based camera to the sampling rate of the external camera system. To analyze the effect of number of events to the pose relocalization results, we perform the following study: During the testing phase using the random split strategy, instead of using all 100% events from two continuous timestamps, we gradually use only 10%, 20%, ..., 90% of these events to create the event images (the events are chosen in order from the current timestamp to the previous timestamp). Fig. 5 shows the position and orientation errors of our SP-LSTM network in this experiment. From the figure, we notice that both the position and orientation errors of all sequences become consistent when we use around 60% number of events. When we use more events to create the event images, the errors are slightly dropped but not significantly. This suggests that the SP-LSTM network still performs well when we use fewer events. We also notice that our current method to create the event image from the events is fairly simple since some of the events may be overwritten when they occur at the same coordinates but have different polarity values with the previous events. Despite this, our SP-LSTM network still successfully relocalizes the camera pose from the event images.

6. Conclusions and Future Work

In this paper, we introduce a new method to relocalize the 6DOF pose of the event camera with a deep network. We first create the event images from the event stream. A deep convolutional neuron network is then used to learn features from the event image. These features are reshaped and fed to a Stacked Spatial LSTM network. We have demonstrated that by using the Stacked Spatial LSTM network to learn spatial dependencies in the feature space, the pose relocalization results can be significantly improved. The experimental results show that our network generalizes well under challenging testing strategies and also gives reasonable results when fewer events are used to create event images. Furthermore, our method has fast inference time and needs only the event image to relocalize the camera pose.

Currently, we employ a fairly simple method to create the event image from a list of events. Our forming method does not check if the event at the local pixel has occurred or not. Since the input of the deep network is the event images, better forming method can improve the pose relocalization results, especially on the cluttered scenes since the data from event cameras are very disorder. Although our network achieves $5ms$ inference time, which can be considered as real-time performance as in PoseNet, it still may not fast enough for high-speed robotic applications using event cameras. Therefore, another interesting problem is to study the compact network architecture that can achieve competitive pose relocalization results while having fewer layers and parameters. This would improve the speed of the network and allow it to be used in more realistic scenarios.

References

[1] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. 7

[2] C. Brandli, R. Berner, M. Yang, S. C. Liu, and T. Delbruck. A 240x180 130 db 3us latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 2014. 1, 2

[3] A. Censi and D. Scaramuzza. Low-latency event-based visual odometry. In *ICRA*, 2014. 1

[4] Keunwoo Choi, George Fazekas, Mark B. Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. *CoRR*, abs/1609.04243, 2016. 3

[5] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *CVPR*, 2017. 2, 5

[6] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. J. Douglas, and T. Delbruck. A pencil balancing robot using a pair of aer dynamic vision sensors. In *International Symposium on Circuits and Systems (ISCAS)*, 2009. 2

[7] Thanh-Toan Do, Trung Pham, Ming Cai, and Ian Reid. Real-time monocular object instance 6d pose estimation. In *BMVC*, 2018. 2

[8] G. Gallego and D. Scaramuzza. Accurate angular velocity estimation with an event camera. *RA-L*, 2017. 2, 3

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computing*, 1997. 3

[10] Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *ICRA*, 2016. 2, 3, 5, 6, 7

[11] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *CVPR*, 2017. 2, 4

[12] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 2, 3, 4, 5, 6, 7

[13] Hanme Kim, Ankur Handa, Ryad Benosman, Sio-Hoi Ieng, and Andrew Davison. Simultaneous mosaicing and tracking with an event camera. In *BMVC*, 2014. 2

[14] Hanme Kim, Stefan Leutenegger, and Andrew J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *ECCV*, 2016. 1, 2

[15] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. In *ICCV*, 2017. 2, 5, 6, 7

[16] E. Mueggler, B. Huber, and D. Scaramuzza. Event-based, 6-dof pose tracking for high-speed maneuvers. In *IROS*, 2014. 1, 2

[17] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbrück, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *IJRR*, 2017. 1, 2, 3, 4

[18] Tayyab Naseer and Wolfram Burgard. Deep regression for monocular camera-based 6-dof global localization in outdoor environments. In *IROS*, 2017. 2

[19] Anh Nguyen, Thanh-Toan Do, Ian Reid, Darwin G Caldwell, and Nikos G Tsagarakis. Object captioning and retrieval with natural language. *arXiv preprint arXiv:1803.06152*, 2018. 2

[20] Anh Nguyen, Thanh-Toan Do, Ian Reid, Darwin G. Caldwell, and Nikos G. Tsagarakis. V2cnet: A deep learning framework to translate videos to commands for robotic manipulation. *arXiv preprint arXiv:1903.10869*, 2019. 2

[21] Henri Rebecq, Timo Horstschaefer, and Davide Scaramuzza. Real-time visualinertial odometry for event cameras using keyframe-based nonlinear optimization. In *BMVC*, volume 3, 2017. 2, 5

[22] Christian Reinbacher, Gottfried Munda, and Thomas Pock. Real-Time Panoramic Tracking for Event Cameras. In *International Conference on Computational Photography (ICCP)*, 2017. 2

[23] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014. 4

[24] Lucas Theis and Matthias Bethge. Generative image modeling using spatial lstms. In *NIPS*. 2015. 3

[25] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep auxiliary learning for visual localization and odometry. In *ICRA*, 2018. 2, 5

[26] Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *ICCV*, 2017. 2, 5, 6, 7

[27] Peng Wang, Ruigang Yang, Binbin Cao, Wei Xu, and Yuanqing Lin. Dels-3d: Deep localization and segmentation with a 3d semantic map. In *CVPR*, 2018. 2

[28] D. Weikersdorfer, D. B. Adrian, D. Cremers, and J. Conradt. Event-based 3d slam with a depth-augmented dynamic vision sensor. In *ICRA*, 2014. 1, 2

[29] Jian Wu, Liwei Ma, and Xiaolin Hu. Delving deeper into convolutional neural networks for camera relocalization. In *ICRA*, 2017. 2

[30] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based visual inertial odometry. In *CVPR*, 2017. 2, 5