Preprints.org

Article

Not peer-reviewed version

TactiGraph: An Asynchronous Graph Neural Network for Contact Angle Prediction Using Neuromorphic Vision-Based Tactile Sensing

Hussain Sajwani , Abdulla Ayyad , Yusra Alkendi * , Mohamad Halwani , Yusra Abdulrahman , AbdelQader Abusafieh , Yahya Zweiri

Posted Date: 10 April 2023

doi: 10.20944/preprints202304.0133.v1

Keywords: tactile sensing; vision-based tactile sensing; event-based vision; robotic manufacturing

Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

TactiGraph: An Asynchronous Graph Neural Network for Contact Angle Prediction Using Neuromorphic Vision-Based Tactile Sensing

Hussain Sajwani [1,2], Abdulla Ayyad [2], Yusra Alkendi [3,*], Mohamad Halwani [2], Yusra Abdulrahman [2,3], Abdulqader Abusafieh [2,4], and Yahya Zweiri [2,3]

1 UAE Ministry of Defense, National Service & Reserve Authority.
2 Advanced Research and Innovation Center. (ARIC); Khalifa University, Abu Dhabi, United Arab Emirates.
3 Department of Aerospace Engineering, at Khalifa University, Abu Dhabi, United Arab Emirates.
4 Research and Development, Strata Manufacturing PJSC, Al Ain, UAE.
* Correspondence: yusra.alkendi@ku.ac.ae

Abstract: Vision-based tactile sensors (VBTS) have become the *de facto* method of giving robots the ability to obtain tactile feedback from their environment. Unlike other solutions to tactile sensing, VBTS offers high spatial resolution feedback without compromising on instrumentation costs or incurring additional maintenance expenses. However, conventional cameras used in VBTS have a fixed update rate and output redundant data, leading to computational overhead downstream. In this work, we present a neuromorphic vision-based tactile sensor (N-VBTS) that employs observations from an event-based camera for contact angle prediction. Particularly, we design and develop a novel graph neural network, dubbed TactiGraph, that asynchronously operates on graphs constructed from raw N-VBTS streams exploiting their spatiotemporal correlations to perform predictions. Although conventional VBTS uses an internal illumination source, TactiGraph is reported to perform efficiently in both scenarios, with and without an internal illumination source. Rigorous experimental results revealed that TactiGraph achieved a mean absolute error of 0.62° in predicting the contact angle and was faster and more efficient than both conventional VBTS and other N-VBTS, with lower instrumentation costs. Specifically, N-VBTS requires only 5.5% of the compute-time needed by VBTS when both are tested on the same scenario.

Keywords: tactile sensing; vision-based tactile sensing; event-based vision; robotic manufacturing

1. Introduction

1.1. *Sense of touch and vision-based tactile sensing*

The sense of touch is an important feedback modality that allows humans to perform many tasks. Consider, for example, the task of inserting a key into a lock. After obtaining an estimate of the keyhole's position, we rely almost exclusively on the sense of touch to get the key from being in the general vicinity of the keyhole to inserting it in the keyhole. During these fleeting seconds, we rely on tactile feedback to adjust both the position as well as the orientation of the key until insertion is obtained. More subtly, we also use tactile feedback to adjust how much force is needed to keep the grasped key from slipping. For robots to perform tasks such as grasping [1,2], peg-in-hole insertion [3,4], and many tasks that require dexterity, it becomes paramount that robotic systems have a sense of touch [5–7]. Much work has been done on augmenting robots with an artificial sense of touch [8]. Several tactile sensor conceptions exist within the literature. These include sensors based on transduction (capacitive, resistive, ferromagnetic, optical, etc.) as well as those based on piezoelectric material [7]. However, these sensors have high instrumentation costs and are thus hard to maintain over long periods of time [9]. A particularly promising tactile sensing technology is vision-based tactile sensors (VBTS) [9].

© 2023 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

VBTS systems consist of an imaging device capturing the deformation of a soft material due to contact with the external environment. The soft material is equipped with a pattern that allows the image sensor to capture deformations clearly. Such patterns include small colored markers [10,11], randomly dispersed fluorescent markers [12], and colored LED patterns [13]. Compared to other tactile sensors, VBTSs do not require much instrumentation; only the imaging device and a source of illumination are required to be instrumented and maintained. This is important for the longevity and usability of the sensor. Although VBTSs have low instrumentation overhead, they also provide high-resolution tactile feedback. VBTS output images, and thus can be processed using classical and learning methods for image processing. Some algorithms build on intermediate features such as marker position, optical flow, or depth [3,10,11,14–18], while others build and train end-to-end models [3,18–20]. The high resolution of VBTS allows robots to perform tasks such as detecting slip [2,21,22], estimating force distribution [11,23], classifying surface textures [24,25], and manipulating of small and soft objects [26,27] as well as many tasks that require dexterity and fine detailed pose estimation.

In previous work of ours [10,28], we have introduced a multifunctional sensor for navigation and guidance, precision machining, and vision-based tactile sensing. Concretely, the sensor is introduced in the context of the aerospace manufacturing industry where maintaining high-precision while machining is imperative. The proposed sensor configuration is depicted in Figure 1. When the hatch is open, the camera used for VBTS is used to navigate to the desired position. Once the target position is achieved, the hatch is closed and the deburring or drilling process starts. The tactile feedback from the VBTS is used to ensure that the robot is machining while maintaining perfect perpendicularity to the workpiece. The video stream of the imaging device is processed by a convolutional neural network (CNN). Ensuring perpendicularity is crucial when working in the aerospace manufacturing industry [29–31]. Failure to abide by perpendicularity requirements leads to an increase in bending stress and a decrease in fatigue life, thus lowering the reliability of aircraft parts [32,33]. Several other works in the literature use VBTS for contact angle prediction [19,34,35]. Lepora et al. [19] use a CNN on binarized images from the camera to estimate the contact pose. Psomopoulou et al. [34] also use a CNN to estimate the contact angle while grasping an object. Finally, [35] extract the markers' positions using blob detection, then construct a 2D graph with markers as nodes using Delaunay triangulation. The graphs are then processed using a graph neural network to predict contact angle. All of the afromentioned work uses conventional VBTS thus requiring internal illumination as well as having a limited temporal update-rate. Due to advancements in vision sensor technologies, the neuromorphic event-based camera has emerged as a critical tool for achieving accurate visual perception and navigation. This bio-inspired device offers unprecedented capabilities compared to standard cameras, including its asynchronous nature, high temporal resolution, high dynamic range, and low power consumption. Therefore, by utilizing an event camera instead of a standard camera, we can enhance the potential of our previous sensor and improve navigation or machining performance in challenging illumination scenarios. In this work, we use an event-based camera for VBTS. This allows us to obtain less expensive computations, fast update-rate, and relinquishing the need for internal illumination that adds instrumentation complexity.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

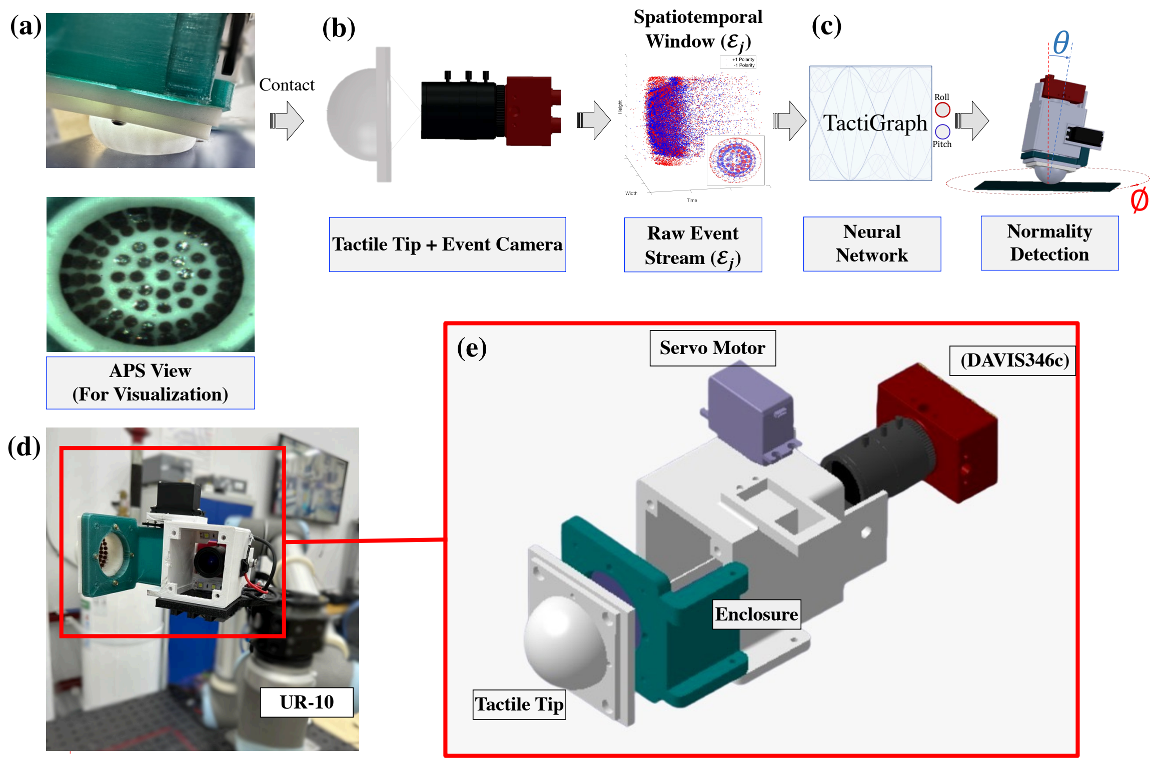

Figure 1. Proposed framework. (a) The neuromorphic vision-based tactile sensor makes contact with the surface causing the tip of the sensor to deform. Markers placed on the inner surface of the tip are displaced due to the deformation of the sensor. This can be seen in the active-pixel sensor (APS) view. This is shown for visualization only and is not used in any further processing. (b) The displacement of the markers is captured by an event-based camera hence generating a continuous stream of events. A volume of events $\mathcal{E}_j$ is used to construct a graph $G_j = (V_j, E_j)$. (c) This graph is processed by TactiGraph, a graph neural network that predicts the contact angle of the sensor. (d) The full sensor with the hatch open. The markers on the inner wall of the tactile tip can be seen. (e) The proposed sensor configuration comprises three main parts: the deformable tactile tip, an event-based camera, and an enclosure holding the camera.

1.2. Neuromorphic vision-based tactile sensing

Most VBTSs, like the ones used in our prior work, use traditional frame-based cameras which record all pixels and return their intensity at a fixed rate. This leads to the redundant processing of pixels that do not change between consecutive frames. The fixed rate of frame-based cameras also poses a problem when recording fast-paced scenes. Furthermore, the downtime between consecutive frames can contain information that is of crucial importance to precision machining. To alleviate these issues, neuromorphic event-based cameras are being employed for vision-based tactile sensing [1,16,20,21,36–39].

Neuromorphic cameras (also known as event-based cameras) are a relatively new technology, first introduced in [40], that aims to mimic how the human eye works. Neuromorphic cameras report intensity changes, at the pixel level, in the scene in an asynchronous manner rather than report the whole frame at a fixed rate. This mode of operation makes event-based cameras exhibit no motion blur. The pixel-wise intensity changes, called events or spikes, are recorded at a temporal resolution on the order of microseconds. Event-based cameras have been applied in autonomous drone racing [41], space imaging [42], space exploration [43], automated drilling [44], and visual servoing [45,46]. Neuromorphic cameras' fast update rate, along with their high dynamic range (140 dB compared to conventional cameras with 60 dB [47]) and low power consumption, make them apt for robotics tasks [48]. In particular, Event-based cameras are capable of providing adequate visual

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

information in challenging lighting conditions without requiring an additional light source, owing to their high dynamic range. Due to not needing a source of illumination, a VBTS system that utilizes an event-based camera will have a lower instrumentation cost and thus require less maintenance in the long run. While some VBTSs use a transparent tactile surface to overcome the need for a source of illumination [9], this will make training end-to-end machine learning models difficult as the camera will capture extraneous information from the environment making it dependent on the object the sensor is contacting and the environment, thus limiting generalization. Event-based cameras allow us to overcome the instrumentation and maintenance costs of having a source of illumination while still maintaining the potential for training end-to-end models. These features of event-based camera make them an attractive choice for VBTS. However, dealing with event-based data still poses a challenge as will be discussed in the following subsection.

1.3. Challenges with event-based vision and existing solutions

The temporally dense, spatially sparse, and asynchronous nature of event-based streams poses a challenge to traditional methods of processing frame-based streams. Early work on neuromorphic vision-based tactile sensing (N-VBTS) constructs images out of event streams by accumulating events over a period of time and applying image-processing techniques. Such approaches are called event-frame methods. These approaches usually use synchronous algorithms and apply them over constructed frames sequentially thus, event-frame approaches do not exploit the temporal density and spatial sparsity of event streams. For instance, Amin et al. [36] detect the incipient slip of a grasped object by applying morphological operations over event-frames and monitoring blobs in the resulting frame. This approach is not asynchronous and does not generalize well to tasks beyond slip detection. Ward-Cherrier et al. [16] construct encodings relevant to the markers' position of the tactile sensors and then use a classifier to detect the object's texture in contact. This approach assumes knowledge of the markers' positioning at all times. To our knowledge, the authors do not provide an algorithm for obtaining the markers' position beyond the case where the sensor is resting. Fariborz et al. [37,38] use Conv-LSTMs on event-frames constructed from event streams to estimate contact forces. Faris et al. [22] uses CNN over accumulated event heatmaps to detect slip. This approach is not asynchronous hence has downtime between constructed event-frames. To our knowledge, the only asynchronous deep learning method that makes use of spatial sparsity and temporal density applied in the N-VBTS setting is the work of MacDonald et al. [39]. Building on [16]'s NeuroTac, they propose using a spiking neural network (SNN) to determine the orientation of contact with an edge. While this is a promising step towards neuromorphic tactile sensing, the training of the SNN is done in an unsupervised manner. Another classifier is run on top of the SNN to make the prediction. This approach does not generalize well beyond simple tasks. Furthermore, training SNNs is still challenging due to their non-differentiable nature [49,50].

Outside the N-VBTS literature, event-frame and voxel methods also persist [47,51–54]. An emerging line of research investigates the use of graph neural networks (GNNs) to process event streams [55–59]. GNNs operate on graphs by learning a representation that takes into account the graph's connectivity. This representation can be used for further processing via classical machine and deep learning methods. GNNs generalize convolutional networks for irregular grids and networks [60]. This is done using a mechanism called *message passing*. Given a graph G with nodes V, node features X, and edges E, a single layer of message passing will have nodes obtaining messages from their neighboring nodes and using those messages to update their node features. These messages are learnable functions of the neighboring nodes' features. A particular type of message passing is the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

convolutional type, where nodes update their representation in a way that resembles convolutions. A node $u \in V$ updates its own representation from x_u^ℓ at layer ℓ, to $x_u^{\ell+1}$ at layer $\ell + 1$ by

$$m_{uv} = a_{uv}\varphi(x_i^\ell, x_j^\ell) \tag{1}$$

$$m_u = \sum_{v\in\mathcal{N}(u)} m_{uv} \tag{2}$$

$$x_u^{\ell+1} = \psi(x_i^\ell, m_i) \tag{3}$$

Where $v \in V$ is a neighboring node of u, a_{uv} are edge attributes, and φ and ψ are learnable functions [60]. By constructing a graph over events from an event-based camera, GNNs can perform spatially sparse and temporally dense convolutions. GNNs can also operate in an asynchronous mode by applying the methods proposed in [61,62] to match the nature of event-based streams. This mode of operation ensures that calculations only occur when there are events, as opposed to event-frame methods. The earliest work utilizing GNNs for event streams, [55], investigates object classification on neuromorphic versions of popular datasets such as Caltech101 and MNIST. Other works also tackle object detection and localization [55,56,61]. Alkendi et al. [59] use a GNN fed into a transformer for event stream denoising. Furthermore, [63] shows that GNNs do well in object detection tasks while performing considerably fewer floating point operations per event compared to CNNs operating on event-frames.

Graphs inherently do not encode geometric information pertaining to their nodes. They only encode information concerning the topological relationships between the nodes as well as the node features. Nonetheless, constructing useful and meaningful representations of event data requires more than just the topological structure of a graph. Thus, it becomes imperative to choose ψ and φ in a way that encapsulates the geometry of events. Several graph geometric deep learning methods have been applied to event-based data in the literature. This includes Mixture model network (MoNet), graph convolutional networks (GCN), SplineConv, voxel graph CNNs, as well as EventConv [55–59]. Although all of these methods take geometry into account, SplineConv has been shown to perform better and faster than MoNet [57]. Furthermore, unlike voxel graph CNNs and EventConv [58,59], SplineConv can be run asynchronously using the method proposed in [63]. SplineConvs are also more expressive than GCNs which can only use one-dimensional features [64,65]. In the case of geometric graphs, this feature is usually taken as the distance between nodes. This is problematic for two reasons 1) messages shared from two equidistant nodes will be indistinguishable and 2) the messages will be rotation invariant and hence lose all information about orientation.

1.4. Contributions

In this work, we use a SplineConv-based graph neural network to predict the contact angle of a neuromorphic vision-based tactile sensor. This proposed framework is depicted in Figure 1. Our contributions can be summarized as follows.

- We introduce, TactiGraph, a graph neural network based on SplineConv layers to handle data from a neuromorphic vision-based tactile sensor. TactiGraph is able to fully account for the spatially sparse and temporally dense nature of event streams.
- We deploy TactiGraph to solve the problem of contact angle prediction using the neuromorphic tactile sensor. We obtain an error of 0.62°.
- TactiGraph maintains a high level of accuracy, with only an error of 0.71°, even when the tactile sensor is used without an illumination source. This demonstrates that it is possible to reduce the cost of operating a vision-based tactile sensor by eliminating the need for LEDs, while still achieving reliable results.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

1.5. Outline

The rest of this work is organized as follows. In section 2, we discuss the data collection apartus as well as the sensor design. In section 3, we benchmark TactiGraph against other VBTS and N-VBTS methods. Finally, we conclude in section 4.

2. Materials and Methods

In this section, we describe the experimental setup used to generate the data in this paper. Furthermore, we also describe the tactile sensor design, the sensor's output format, as well as how to handle this output using TactiGraph.

2.1. Data collection

The VBTS system consists of a camera, an enclosure containing the camera, and a hemispherical soft surface attached to the hatch as seen in Figure 1. Small beads are placed on the insides of the soft tactile surface to generate contrast, allowing the camera to capture the deformation of the sensor with clarity. The event-camera used is an IniVation DAVIS 346 with resolution 346 $\times$ 260 and a latency of $20\mu s$ [66]. The sensor enclosure is made of 3D-printed ABS. Two LED strips are placed above and below the camera. More details on the manufacturing process of the elastomer surface and the enclosure can be found in [10,28]. The whole apparatus is attached to the Universal Robotics UR10 [67]. The UR10 pushes the tactile sensor against a flat surface at various angles of contact. This is shown in Figure 1. The contact angle is controlled by two parameters, $\theta \in \Theta$ and $\phi \in \Phi$ where $\Theta = \{0, 1, \ldots, 9\}$ and Φ is a collection of 20 angles around the circle. This variation in θ and ϕ can be seen in Figure 1. We collect 12 samples of each contact angle, thus ending up with a total of $n = 1836$ samples in total. The depth of each contact case is chosen randomly between 5mm and 15mm from the tip of the sensor when relaxed. The randomness in contact depth ensures that our model can generalize to different contact profiles from light to heavy contact cases. To evaluate the performance of N-VBTS without internal illumination, this process is done twice, once with the LED strips on and another time with the LED strips turned off.

2.2. Preprocessing the event stream

Let $C = \{1, \ldots, 240\} \times \{1, \ldots, 346\}$ the set of all pixels in the DAVIS 346. The output of the event camera is then a stream $\mathcal{S} = \{e_k\}_{k=1}^{N}$ of asynchronously reported intensity changes in the frame. The k^{th} intensity change, an event, is a 4-tuple

$$e_k = (x_k, y_k, t_k, p_k) \tag{4}$$

where $(x_k, y_k) \in C$, $t_k \in \mathbb{R}^+$, and $p_k \in \{-1, 1\}$. The (x, y) components represent where the event has happened, the t component is a timestamp indicating when the event has happened, and p, the polarity, represents whether the intensity change is positive or negative. Figure 2 shows histograms of the number of events generated by contact cases in both LED-on and LED-off scenarios. The variance in the depth of the contact translates itself into the variance of the number of events generated; a light contact case will cause a smaller displacement to the markers hence generating less events and vice versa. Overall, fewer events are generated in LED-off contact cases. However, dark scenes cause a lot of background noise [68,69]. While many denoising methods exist in the literature [59,68,69], we use the background activity filter [70] with a time window of 1.25 milliseconds when the LED strips are off. For further robustness against noise, the dataset is augmented via jittering events. Given an event $e = (x, y, t, p)$, we spatially jitter the event by at most one pixel $\tilde{e} = (x + \delta x, y + \delta y, t, p)$ where δx and δy are drawn uniformly from {-1, 0, 1}. The effect of this jittering will be investigated later in this work.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

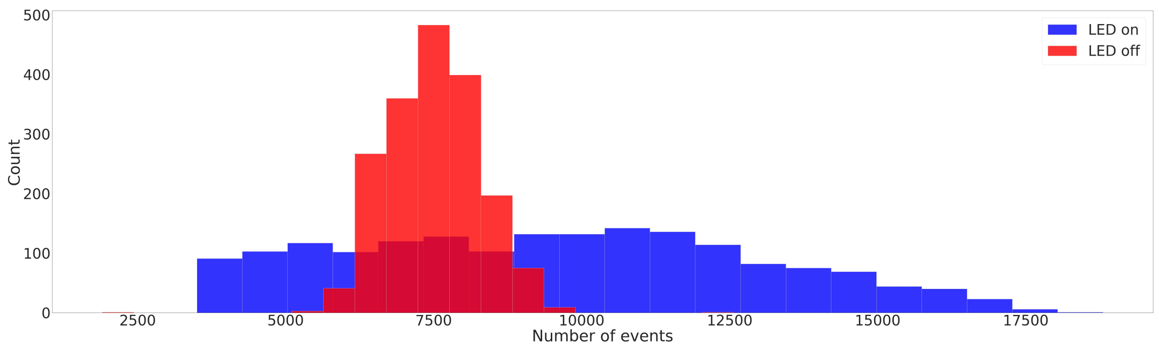

Figure 2. Histograms showing the number of events resulting from each contact case. We plot the LED-off and LED-on dataset's histograms.

Out of the stream $\mathcal{S}$, for the j^{th} contact case we capture a spatiotemporal volume $\mathcal{E}_j \subset \mathcal{S}$ such that

$$\mathcal{E}_j = \{(x, y, t, p) \in \mathcal{S} \mid t_j \leq t \leq t_j + \Delta T_j\} \tag{5}$$

where t_j the timestamp at which the j^{th} contact case starts and $90\text{ms} \leq \Delta T_j \leq 200\text{ms}$ is the window size chosen to be at most 200ms. The window size is adaptive to adjust for various depths of contact cases; a heavy contact case takes more time than a light contact. For each event $e_k \in \mathcal{E}_j$ we assign a normalized spatiotemporal postion $r_k = (\hat{x}_k, \hat{y}_k, \hat{t}_k) = \left(\frac{x}{W}, \frac{y}{H}, \frac{t_k - t_j}{\Delta T_j}\right)$.

2.3. Graph construction

For each spatiotemporal volume of events $\mathcal{E}_j$ a graph $G_j = (V_j, E_j, X_j)$ is construced. The nodes of the graph are events themselves $V_j = \mathcal{E}_j$. The edges of the graph are determined using the k^{th} nearest neighbors algorithm. The distance between two events $e_i, e_k \in \mathcal{E}_j$ is calculated using the Euclidean distance between their normalized spatiotemporal positions r_i and r_k. Letting $k\text{NN}(e_k)$ denote the set of the k nearest events to e_k, the set of edges of the graph is defined by

$$E_j = \{(e_i, e_k) \in \mathcal{E}_j \times \mathcal{E}_j \mid e_k \in k\text{NN}(e_i)\} \tag{6}$$

While this method of constructing the graph does not always result in a graph with one connected component, we found it always results in one large connected component with a few smaller components consisting mostly of less than 10 events. Thus these smaller components are dropped out from the dataset. Finally, each node of the graph e_k has the polarity p_k as its node features, $X_j = \{p_k \mid e_k \in \mathcal{E}_j\}$. The dataset is now the collection of the n graphs, with labels correponding to the contact angle of the contact case, collected from each of the n contact case, $\mathcal{D}_{\text{LED-on}} = \{(G_j, \text{Roll}_\text{j}, \text{Pitch}_j)\}_{j=1}^{n}$. We note that we use a roll-pitch representation of the contact angle. This is to avoid singularities caused around 0. Another dataset $\mathcal{D}_{\text{LED off}}$ with $n = 1836$ samples is generated identically while the LED is off. The two datasets are used separately. Each dataset is randomly split into train, validation, and test subsets with 70% of $\mathcal{D}$ being used for training while 15% is used for validation, and the last 15% is used for testing.

2.4. TactiGraph

A graph neural network, namely TactiGraph, with a stack of SplineConv layers and node pooling layers is used for contact angle prediction. TactiGraph consists of SplineBlocks, node pooling layers, a final pooling layer, and a final multilayer perceptron (MLP). Graphical depiction of TactiGraph can be seen in Figure 3 (a). SplineBlocks consist of a SplineConv layer, an exponential linear unit (ELU), and a batch normalization layer. The node pooling layer reduces the number of nodes in the graph G_j from N_i to N_{i+1} by first constructing a voxel grid over the volume $\mathcal{E}_j$ then pooling all nodes within

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

a voxel unit into one node; inheriting edges, if any, from the unpooled nodes. For the layers before pooling, messages passed between events. This ensures the initial low-level geometry is preserved. After pooling, higher-level nodes are created and message passing occurs between high-level features. The final pooling layer will convert the variable size graph into a fixed-size vector by max pooling over a 4×4 voxel grid. A voxel grid is generated over the spatiotemporal volume $\mathcal{E}$. Skip connections are also added via adding node features. We ablate over the number of SplineConv layers, the number of node pooling layers, the number of skip layers, and the node embedding dimension. The scope of this ablation study is shown in Table 1. A final pooling layer is applied, followed by fully connected layers predicting the contact angles.

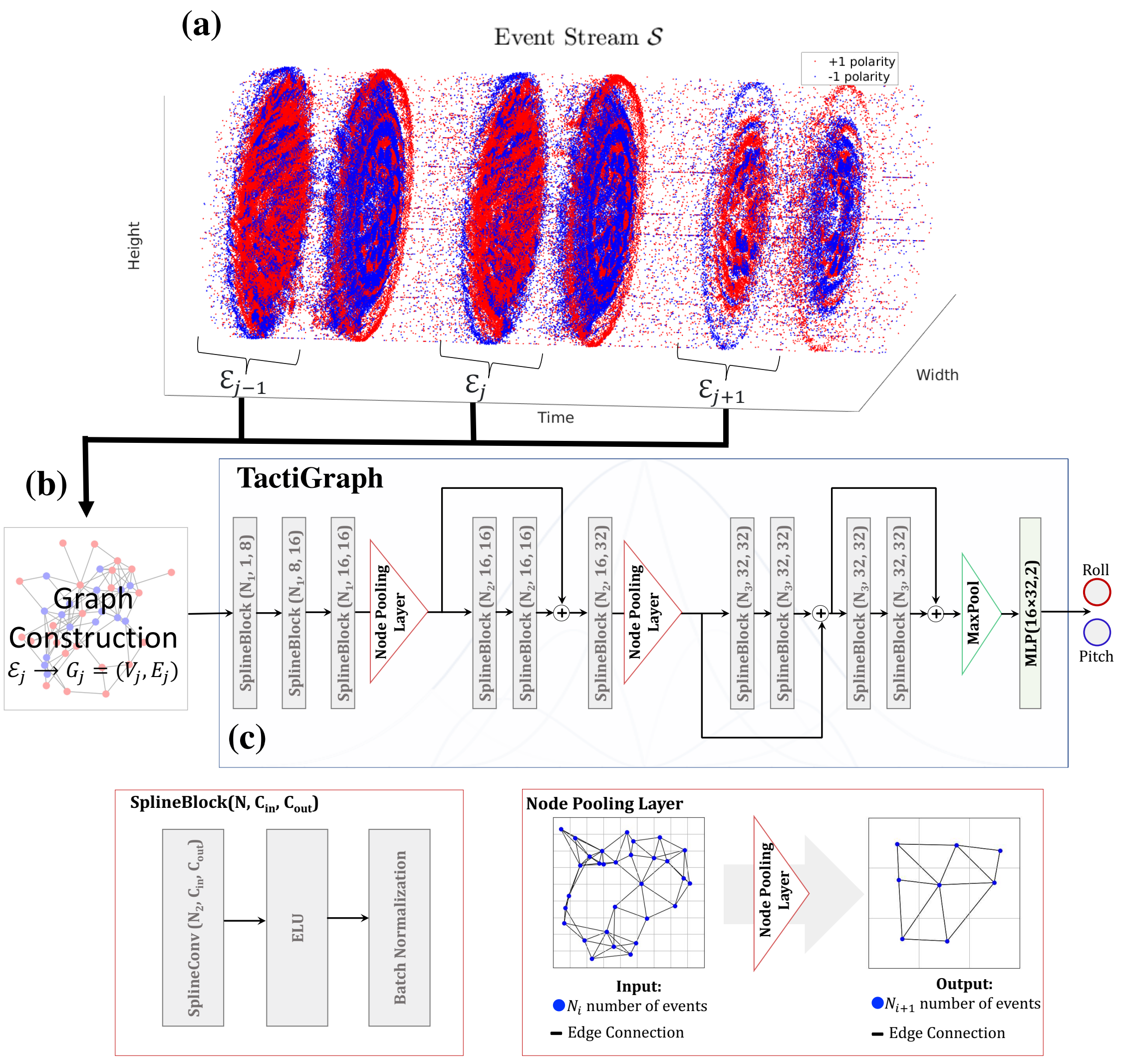
(a)
Event Stream $\mathcal{S}$
+1 polarity
-1 polarity
Height
Width
Time
$\mathcal{E}_{j-1}$
$\mathcal{E}_j$
$\mathcal{E}_{j+1}$
(b)
Graph Construction
$\mathcal{E}_j \rightarrow G_j = (V_j, E_j)$
TactiGraph
SplineBlock (N$_1$, 1, 8)
SplineBlock (N$_1$, 8, 16)
SplineBlock (N$_1$, 16, 16)
Node Pooling Layer
SplineBlock (N$_2$, 16, 16)
SplineBlock (N$_2$, 16, 16)
SplineBlock (N$_2$, 16, 32)
Node Pooling Layer
SplineBlock (N$_3$, 32, 32)
SplineBlock (N$_3$, 32, 32)
SplineBlock (N$_3$, 32, 32)
SplineBlock (N$_3$, 32, 32)
MaxPool
MLP(16×32,2)
Roll
Pitch
(c)
SplineBlock(N, C$_{in}$, C$_{out}$)
SplineConv (N$_2$, C$_{in}$, C$_{out}$)
ELU
Batch Normalization
Node Pooling Layer
Node Pooling Layer
Input:
● N_i number of events
Edge Connection
Output:
● N_{i+1} number of events
Edge Connection

Figure 3. (a) The temporally dense and spatially sparse raw event stream $\mathcal{S}$ resulting from the data collection setup. Each one of the circles in the stream corresponds to a compression or retraction of the tactile sensor. Inside these circles, traces of the markers' movement can be seen. Spatiotemporal volumes $\mathcal{E}_j$ are extracted. (b) Out of each spatiotemporal volume $\mathcal{E}_j$, a graph G_j is constructed. (c) The graphs are fed into TactiGraph whose architecture is shown.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

Table 1. Hyperparameter search for SplineConv graph neural network.

Hyperparameter	Hyperparameter range	Optimal value
Number of SplineConv layers	{6, 7, 8, 9, 10, 11}	10
Number of channels in layers	{8, 16, 32, 64, 128}	(8, 8, 16, 16, 16, 16, 32, 32, 32, 32)
Number of pooling layers	{1, 2, 3}	3
Number of skip connections	{0, 1, 2, 3, 4}	3

2.5. Training setup

We use version 2.0.4 of the Pytorch Geometric library [71] to implement our models. Training is done over 1000 epochs using the Adam optimizer [72] with an adaptive learning rate and default values as per Pytorch version 1.11.0 [73]. The learning rate starts at 0.001 but is reduced to 0.00001 when the validation loss plateaus. The training is carried out on a PC running Ubuntu 20.04, with an Intel i7-12700H CPU and an NVIDIA RTX 3080 Laptop GPU.

3. Results and Discussion

In this section, we present our findings in benchmarking TactiGraph against other methods of contact angle prediction as well as other methods of processing event streams. We demonstrate the abilities of TactiGraph on N-VBTS with and without internal illumination. We conduct a computational analysis comparing TactiGraph on N-VBTS with conventional VBTS approaches.

3.1. Evaluation of TactiGraph and the effect of jittering

The best model from the ablation study is shown in Figure 3. The training results on both LED-on and LED-off datasets are shown in Table 2. We display TactiGraph's mean absolute error (MAE) on the test dataset. The model is trained with and without applying the 1px jitter augmentation on the training datasets. We note that applying the jittering augmentation strategy when training improves accuracy in the test dataset. The effect of jittering is amplified on the LED-off dataset. We argue that this is due to the noisy and sparse nature of event-based cameras in the LED-off environment [68,69]. Thus exposing the model to jittered events makes the model more robust to the noisy test dataset. It is worth noting that jittering events by more than one pixel proved to be an ineffective strategy that gave worse results than not jittering. This might be due to the fact that the event-based camera used, the DAVIS346, is of relatively low resolution. Thus jittering by more than one pixel can cause a change in the true dynamics of the scene.

Table 2. MAE of training TactiGraph on the dataset with the LED-on and the dataset with the LED-off. We compare results before and after adding jittering augmentation.

Dataset used	MAE Before jittering	MAE After jittering
$\mathcal{D}_{\text{LED-on}}$	0.65°	**0.63°**
$\mathcal{D}_{\text{LED off}}$	0.81°	**0.71°**

3.2. Visualizing TactiGraph's embedding space

To get a better understanding of what TactiGraph has learned, we visualize the node embedding generated for each sample in the dataset. This is done by saving the values obtained from the last node pooling layer, right before the fully connected layer, during a forward pass on the trained model. These values live in a high-dimensional space. To this end, we use the t-distributed stochastic neighbor embedding (t-SNE) algorithm [74]. t-SNE, a manifold learning technique, nonlinearly projects high-dimensional data onto a low-dimensional space more tangible for visualization. The results of the t-SNE visualization are shown in Figure 4. Each point in the scatter plot represents a contact case j

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

associated with $(G_j, \text{Roll}_j, \text{Pitch}_j) \in \mathcal{D}_{\text{LED on}}$. The points are colored according to the angles of contact θ and ϕ. Even though TactiGraph was trained on roll-pitch representation, what we see in these plots is that TactiGraph has learned to embed similar contact angles θ and ϕ next to each other. Looking at how different values of ϕ and θ vary in the embedding space, we see that the model has learned an embedding that emphasizes variance in ϕ. This is due to the fact that ϕ varies more than θ. The clearly visible gradients in these plots confirm that TactiGraph has actually done a good job of learning the dynamics of the problem.

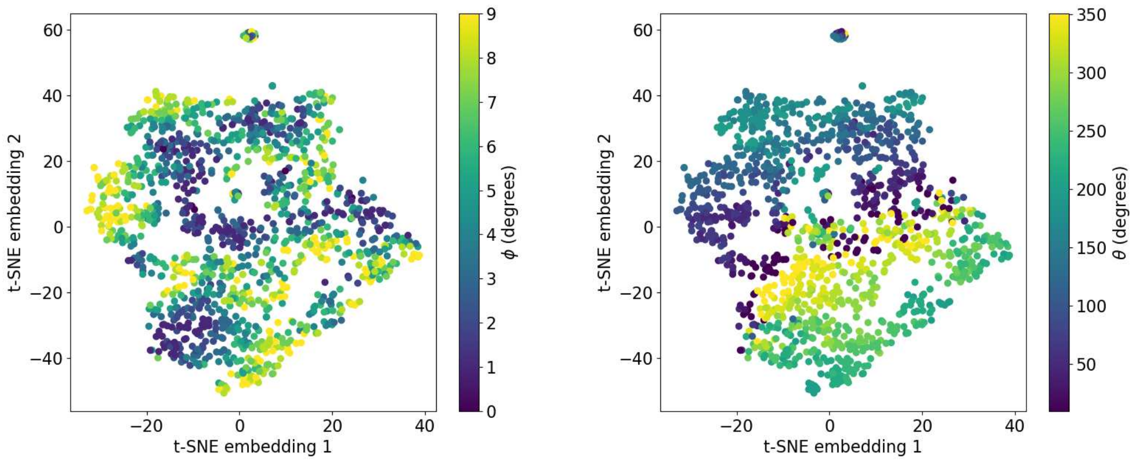

Figure 4. *t*-SNE embeddings of the activations of TactiGraph's last layer before the MLP. Each point is the embedding of one graph G_j corresponding to j^{th} contact case. *Left* Colored by the ϕ angle of the contact case. *Right* Colored by the ϕ angle of the contact case.

3.3. Benchmark results

The only contact angle estimation approach in the literature that uses neuromorphic vision-based tactile sensing is the work of MacDonald et al. [39]. However, unlike our work, MacDonald et al. estimate the contact angle with an edge rather than a flat surface; thus these results are not directly comparable. They build an embedding using a spiking neural network in an unsupervised fashion, which is coupled with a supervised KNN classifier. We also compare our results against other works using traditional vision-based tactile sensing approaches. The results are tabulated in Table 3. We split the results between N-VBTS methods and conventional VBTS methods. Given the relatively low dynamic range of conventional cameras, conventional VBTSs do not work without an internal source of illumination. While Halwani et al. [28] achieve better results, this is done using a CNN operating on a conventional camera, hence requiring a source of illumination, being susceptible to motion blur, as well as being computationally more expensive as we show in section 3.4. The same applies to Tac-VGNN [35], which uses a GNN. However, their GNNs operate synchronously on graphs constructed using a conventional camera, where graph nodes are made of internal markers.

Table 3. MAE of TactiGraph compared to VBTS methods from the literature.

	Neuromorphic VBTS		VBTS	
Internal illumination[‡]	TactiGraph	MacDonald et al.[39][†]	Halwani et al. [28]	Tac-VGNN [35]
With illumination	0.63°	4.44°	0.13°	1.02°
Without illumination	0.71°	-	-	-

[†] Contact is not made against a flat surface.
[‡] For TactiGraph results, the datasets used for different illumination conditions are $\mathcal{D}_{\text{LED on}}$ and $\mathcal{D}_{\text{LED off}}$ as described in section 2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

Finally, we compare the performance of TactiGraph against using CNNs on event-frames. We use each $\mathcal{E}_j$ from both LED-on and LED-off datasets to construct an event-frame. This is done to make a fair comparison, as both methods will have access to the same amount of information. A grayscale event-frame $F_j \in \mathbb{R}^{346\times 260}$ corresponding to volume $\mathcal{E}_j$ is constructed as follows.

$$(F_j)_{ab} = \sum_{e_i \in \mathcal{E}_j} p_i \delta(x_i = a)\delta(y_i = b), \tag{7}$$

where δ is the Dirac delta function. Sample event-frames are shown in Figure 5. We see that the temporal information is lost. Starting from the best CNN model obtained for VBTS contact angle prediction in [28], we perform an ablation study over the number of channels in the 2nd, 3rd, and 4th convolutional layers. Furthermore, we also look at how many convolutional layers are needed. Lastly, the ablation also studies the number and sizes of the fully connected layers. This ablation study is presented in Table 5. The benchmark results of the best CNN model are displayed in Table 4. TactiGraph achieves better contact angle estimation than the optimal CNN architecture on event-frames constructed from the same volumes of events that the graphs are made of. Better results hold in both LED-on and LED-off scenarios.

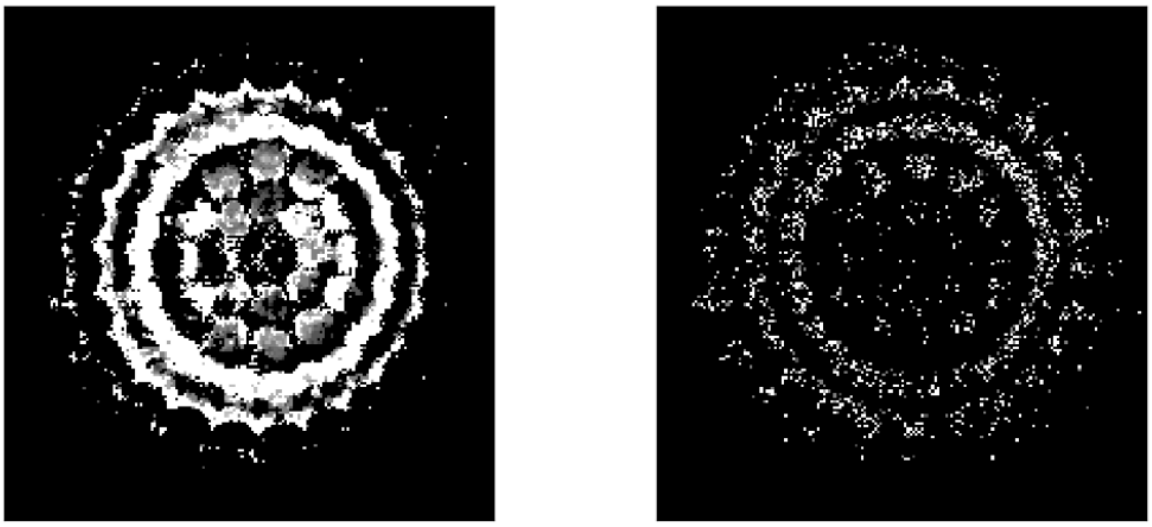

Figure 5. *Left:* sample event-frame constructed from the LED-on dataset. *Right:* sample event-frame constructed from the LED-off dataset.

Table 4. MAE of TactiGraph compared to CNN on event-frame.

Dataset used	TactiGraph	CNN on event-frame
$\mathcal{D}_{\text{LED on}}$	**0.63°**	0.79°
$\mathcal{D}_{\text{LED off}}$	**0.71°**	0.80°

Table 5. Hyperparameter search for a convolutional neural network on event-frames. The architecture with the lowest error is used for comparison against TactiGraph.

Hyperparameter	Hyperparameter range	Optimal value
Number of convolutional layers	{3,4,5,6,7}	6
Number of channels in layers	{16,32,64,128,256}	(32, 32, 32, 128, 128, 256)
Number of dense layers	{2,3,4,5}	4

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

3.4. Runtime analysis

Given a live event stream $\mathcal{S}$ that started running at time t_0, in other words, for every $(x, y, t, p) \in \mathcal{S}$ we have $t_0 \leq t \leq t_c$ where t_c is the current time, TactiGraph operates on a graph constructed from a sliding window.

$$\mathcal{W} = \{(x, y, t, p) \in \mathcal{S} \,|\, t_c - \Delta T \leq t \leq t_c\} \tag{8}$$

As events asynchronously enter and exit $\mathcal{W}$, the graph is updated accordingly and TactiGraph acts on it. Instead of having a GNN rerun the whole forward pass as events slide in and out of $\mathcal{W}$, Schaefer et al. [61] propose AEGNN, a method by which redundant computations are not repeated. By looking at the neighborhoods of incoming and outcoming events, AEGNN is able to asynchronously determine which computations need to be recomputed. We modify TactiGraph to utilize the same mechanism as AEGNN. With these optimizations in mind, a prediction using TactiGraph consists of two steps: graph construction and the forward pass. In the worst-case scenario where the whole scene changes, the graph construction step takes an average of 34.5 milliseconds. Also, in the worst-case scenario, the forward pass takes an average of 58.1 milliseconds. These results were obtained on the same hardware mentioned above in section 2.5.

The combination of N-VBTS and TactiGraph is computationally much cheaper than the CNN and VBTS of Halwani et al. [28]. We validate this by looking at the total compute-time taken by both methods in processing the same contact cases. We record a dataset 20 seconds long containing five contact cases. We run the CNN model from [28] on the active pixel sensor stream of the same DAVIS 346 used in this work. The total compute time the CNN takes to process this stream is 3.93 seconds. TactiGraph operating on the event stream, on the other hand, took only 0.22 seconds, 5.5% of the CNN compute time. This is attributed to the redundant output the VBTS gives, which leads to redundant computations done by the CNN. Therefore, TactiGraph operating on N-VBTS streams is much faster than the CNN model operating on VBTS streams from [28].

3.5. Future work

Our neuromorphic vision-based tactile sensor has shown remarkable performance in contact angle prediction. Therefore, the TactiGraph capabilities can be extended further to perform other tactile sensing applications such as force sensing, texture recognition, and object identification in parallel. We plan on also including a recurrent or attentional mechanism in TactiGraph. This will give TactiGraph the generalization ability to operate on multiple tasks. It is also worth noting that the forward pass in TactiGraph can be further improved by replacing SplineConvs with a corresponding look-up table as proposed in [63], which claims a 3.7-fold reduction in inference time.

4. Conclusions

We introduced a neuromorphic vision-based tactile sensor (N-VBTS) that is able to run at a faster rate than the traditional vision-based tactile sensor (VBTS). We developed TactiGraph, a graph neural network, to operate on the raw asynchronous event stream exploiting the spatiotemporal correlations between events. Notably, TactiGraph is utilized to predict the contact angle of the sensor and achieves an error of 0.62° degrees. We demonstrated the effectiveness of the proposed N-VBTS in terms of efficacy and accuracy compared to VBTS. Particularly, N-VBTS were capable of functioning without internal illumination hence leading to a reduction in long-term instrumentation and maintenance requirements. When tested on the same scenario, N-VBTS requires only 5.5% of the compute-time needed by VBTS.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

Author Contributions: Conceptualization, H. Sajwani., A. Ayyad and Y. Zweiri.; methodology, H. Sajwani., A. Ayyad., M. Halwani and Y. Alkendi; software, H. Sajwani.; validation, H. Sajwani. and A. Ayyad.; formal analysis, H. Sajwani.; investigation, H. Sajwani.; resources, A. Ayyad, M. Halwani; data curation, H. Sajwani., A. Ayyad., and M. Halwani.; writing—original draft preparation, H. Sajwani., Y. Alkendi; writing—review and editing, H. Sajwani., A. Ayyad, Y. Alkendi, M. Halwani, Y. Abdulrahman, and Y. Zweiri.; visualization, H. Sajwani. and Y. Alkendi; supervision, Y. Zweiri., Y. Abdulrahman and A. Abusafieh.; project administration, Y. Zweiri, Y. Abdulrahman and A. Abusafieh; funding acquisition, Y. Zweiri All authors have read and agreed to the published version of the manuscript.

Funding: This work This work was performed at the Advanced Research and Innovation Center and was funded by STRATA Manufacturing PJSC (a Mubadala company), Sandooq Al Watan under Grant SWARD-S22-015, and Khalifa University of Science and Technology.

Acknowledgments: The authors would like to thank Mohamed Salah and Mohamad Ramadan for their help in calibrating the sensor as well as operating the manipulator.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

VBTS	Vision-based tactile sensing
N-VBTS	Neuromorphic vision-based tactile sensing
LED	Light-emitting diode
GNN	Graph neural network
CNN	Convolutional neural network
SNN	Spiking neural network
MAE	Mean absolute error
kNN	*k*-nearest neighbors
t-SNE	*t*-distributed stochastic neighbor embedding

References

1. Huang, X.; Muthusamy, R.; Hassan, E.; Niu, Z.; Seneviratne, L.; Gan, D.; Zweiri, Y. Neuromorphic Vision Based Contact-Level Classification in Robotic Grasping Applications. *Sensors* **2020**, *20*. doi:10.3390/s20174724.
2. James, J.W.; Pestell, N.; Lepora, N.F. Slip Detection With a Biomimetic Tactile Sensor. *IEEE Robotics and Automation Letters* **2018**, *3*, 3340–3346. doi:10.1109/LRA.2018.2852797.
3. Dong, S.; Jha, D.; Romeres, D.; Kim, S.; Nikovski, D.; Rodriguez, A. Tactile-RL for Insertion: Generalization to Objects of Unknown Geometry. 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.
4. Kim, S.; Rodriguez, A. Active Extrinsic Contact Sensing: Application to General Peg-in-Hole Insertion. 2022 International Conference on Robotics and Automation (ICRA), 2022, pp. 10241–10247. doi:10.1109/ICRA46639.2022.9812017.
5. Xia, Z.; Deng, Z.; Fang, B.; Yang, Y.; Sun, F. A review on sensory perception for dexterous robotic manipulation. *International Journal of Advanced Robotic Systems* **2022**, *19*, 17298806221095974, [https://doi.org/10.1177/17298806221095974]. doi:10.1177/17298806221095974.
6. Li, Q.; Kroemer, O.; Su, Z.; Veiga, F.F.; Kaboli, M.; Ritter, H.J. A Review of Tactile Information: Perception and Action Through Touch. *IEEE Transactions on Robotics* **2020**, *36*, 1619–1634. doi:10.1109/TRO.2020.3003230.
7. Dahiya, R.S.; Valle, M. *Robotic Tactile Sensing*; Springer Netherlands, 2013. doi:10.1007/978-94-007-0579-1.
8. Romeo, R.A.; Zollo, L. Methods and Sensors for Slip Detection in Robotics: A Survey. *IEEE Access* **2020**, *8*, 73027–73050. doi:10.1109/ACCESS.2020.2987849.
9. Shah, U.H.; Muthusamy, R.; Gan, D.; Zweiri, Y.; Seneviratne, L. On the Design and Development of Vision-based Tactile Sensors. *Journal of Intelligent & Robotic Systems* **2021**, *102*, 82. doi:10.1007/s10846-021-01431-0.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

10. Zaid, I.M.; Halwani, M.; Ayyad, A.; Imam, A.; Almaskari, F.; Hassanin, H.; Zweiri, Y. Elastomer-Based Visuotactile Sensor for Normality of Robotic Manufacturing Systems. *Polymers* **2022**, *14*. doi:10.3390/polym14235097.
11. Lepora, N.F. Soft Biomimetic Optical Tactile Sensing With the TacTip: A Review. *IEEE Sensors Journal* **2021**, *21*, 21131–21143. doi:10.1109/JSEN.2021.3100645.
12. Sferrazza, C.; D'Andrea, R. Design, Motivation and Evaluation of a Full-Resolution Optical Tactile Sensor. *Sensors* **2019**, *19*. doi:10.3390/s19040928.
13. Lambeta, M.; Chou, P.W.; Tian, S.; Yang, B.; Maloon, B.; Most, V.R.; Stroud, D.; Santos, R.; Byagowi, A.; Kammerer, G.; Jayaraman, D.; Calandra, R. DIGIT: A Novel Design for a Low-Cost Compact High-Resolution Tactile Sensor with Application to In-Hand Manipulation. *IEEE Robotics and Automation Letters (RA-L)* **2020**, *5*, 3838–3845. doi:10.1109/LRA.2020.2977257.
14. Yuan, W.; Dong, S.; Adelson, E.H. GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force. *Sensors* **2017**, *17*. doi:10.3390/s17122762.
15. Wang, S.; She, Y.; Romero, B.; Adelson, E.H. GelSight Wedge: Measuring High-Resolution 3D Contact Geometry with a Compact Robot Finger. 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021.
16. Ward-Cherrier, B.; Pestell, N.; Lepora, N.F. NeuroTac: A Neuromorphic Optical Tactile Sensor applied to Texture Recognition. 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 2654–2660. doi:10.1109/ICRA40945.2020.9197046.
17. Bauza, M.; Valls, E.; Lim, B.; Sechopoulos, T.; Rodriguez, A. Tactile Object Pose Estimation from the First Touch with Geometric Contact Rendering, 2020. doi:10.48550/ARXIV.2012.05205.
18. Li, M.; Li, T.; Jiang, Y. Marker Displacement Method Used in Vision-Based Tactile Sensors—From 2D to 3D: A Review **2023**. doi:10.36227/techrxiv.22122596.v1.
19. Lepora, N.F.; Lloyd, J. Optimal Deep Learning for Robot Touch: Training Accurate Pose Models of 3D Surfaces and Edges. *IEEE Robotics and Automation Magazine* **2020**, *27*, 66–77. doi:10.1109/MRA.2020.2979658.
20. Faris, O.; Muthusamy, R.; Renda, F.; Hussain, I.; Gan, D.; Seneviratne, L.; Zweiri, Y. Proprioception and Exteroception of a Soft Robotic Finger Using Neuromorphic Vision-Based Sensing. *Soft Robotics* **0**, *0*, null, [https://doi.org/10.1089/soro.2022.0030]. PMID: 36251962, doi:10.1089/soro.2022.0030.
21. Muthusamy, R.; Huang, X.; Zweiri, Y.; Seneviratne, L.; Gan, D. Neuromorphic Event-Based Slip Detection and Suppression in Robotic Grasping and Manipulation. *IEEE Access* **2020**, *8*, 153364–153384. doi:10.1109/ACCESS.2020.3017738.
22. Faris, O.; Alyammahi, H.; Suthar, B.; Muthusamy, R.; Shah, U.H.; Hussain, I.; Gan, D.; Seneviratne, L.; Zweiri, Y. Design and experimental evaluation of a sensorized parallel gripper with optical mirroring mechanism. *Mechatronics* **2023**, *90*, 102955. doi:https://doi.org/10.1016/j.mechatronics.2023.102955.
23. Quan, S.; Liang, X.; Zhu, H.; Hirano, M.; Yamakawa, Y. HiVTac: A High-Speed Vision-Based Tactile Sensor for Precise and Real-Time Force Reconstruction with Fewer Markers. *Sensors* **2022**, *22*. doi:10.3390/s22114196.
24. Li, R.; Adelson, E.H. Sensing and Recognizing Surface Textures Using a GelSight Sensor. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1241–1247. doi:10.1109/CVPR.2013.164.
25. Pestell, N.; Lepora, N.F. Artificial SA-I, RA-I and RA-II/vibrotactile afferents for tactile sensing of texture. *Journal of The Royal Society Interface* **2022**, *19*. doi:10.1098/rsif.2021.0603.
26. Li, R.; Platt, R.; Yuan, W.; ten Pas, A.; Roscup, N.; Srinivasan, M.A.; Adelson, E. Localization and manipulation of small parts using GelSight tactile sensing. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 3988–3993. doi:10.1109/IROS.2014.6943123.
27. She, Y.; Wang, S.; Dong, S.; Sunil, N.; Rodriguez, A.; Adelson, E. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research* **2021**, *40*, 1385–1401, [https://doi.org/10.1177/02783649211027233]. doi:10.1177/02783649211027233.
28. Halwani, M.; Ayyad, A.; AbuAssi, L.; Abdulrahman, Y.; Almaskari, F.; Hassanin, H.; Abusafieh, A.; Zweiri, Y. A Novel Vision-based Multi-functional Sensor for Normality and Position Measurements in Precise Robotic Manufacturing. *SSRN Electronic Journal* **2023**. doi:10.2139/ssrn.4360666.
29. Santos, K.R.d.S.; de Carvalho, G.M.; Tricarico, R.T.; Ferreira, L.F.L.R.; Villani, E.; Sutério, R. Evaluation of perpendicularity methods for a robotic end effector from aircraft industry. 2018 13th IEEE International Conference on Industry Applications (INDUSCON), 2018, pp. 1373–1380. doi:10.1109/INDUSCON.2018.8627218.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

30. Zhang, Y.; Ding, H.; Zhao, C.; Zhou, Y.; Cao, G. Detecting the normal-direction in automated aircraft manufacturing based on adaptive alignment. *Sci Prog* **2020**, *103*, 36850420981212.
31. Yu, L.; Zhang, Y.; Bi, Q.; Wang, Y. Research on surface normal measurement and adjustment in aircraft assembly. *Precision Engineering* **2017**, *50*, 482–493. doi:https://doi.org/10.1016/j.precisioneng.2017.07.004.
32. Lin, M.; Yuan, P.; Tan, H.; Liu, Y.; Zhu, Q.; Li, Y. Improvements of robot positioning accuracy and drilling perpendicularity for autonomous drilling robot system. 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2015, pp. 1483–1488. doi:10.1109/ROBIO.2015.7418980.
33. Tian, W.; Zhou, W.; Zhou, W.; Liao, W.; Zeng, Y. Auto-normalization algorithm for robotic precision drilling system in aircraft component assembly. *Chinese Journal of Aeronautics* **2013**, *26*, 495–500. doi:https://doi.org/10.1016/j.cja.2013.02.029.
34. Psomopoulou, E.; Pestell, N.; Papadopoulos, F.; Lloyd, J.; Doulgeri, Z.; Lepora, N.F. A Robust Controller for Stable 3D Pinching Using Tactile Sensing. *IEEE Robotics and Automation Letters* **2021**, *6*, 8150–8157. doi:10.1109/LRA.2021.3104057.
35. Fan, W.; Yang, M.; Xing, Y.; Lepora, N.F.; Zhang, D. Tac-VGNN: A Voronoi Graph Neural Network for Pose-Based Tactile Servoing, 2023. doi:10.48550/ARXIV.2303.02708.
36. Rigi, A.; Baghaei Naeini, F.; Makris, D.; Zweiri, Y. A Novel Event-Based Incipient Slip Detection Using Dynamic Active-Pixel Vision Sensor (DAVIS). *Sensors* **2018**, *18*. doi:10.3390/s18020333.
37. Baghaei Naeini, F.; AlAli, A.M.; Al-Husari, R.; Rigi, A.; Al-Sharman, M.K.; Makris, D.; Zweiri, Y. A Novel Dynamic-Vision-Based Approach for Tactile Sensing Applications. *IEEE Transactions on Instrumentation and Measurement* **2020**, *69*, 1881–1893. doi:10.1109/TIM.2019.2919354.
38. Naeini, F.B.; Kachole, S.; Muthusamy, R.; Makris, D.; Zweiri, Y. Event Augmentation for Contact Force Measurements. *IEEE Access* **2022**, *10*, 123651–123660. doi:10.1109/ACCESS.2022.3224584.
39. Macdonald, F.L.A.; Lepora, N.F.; Conradt, J.; Ward-Cherrier, B. Neuromorphic Tactile Edge Orientation Classification in an Unsupervised Spiking Neural Network. *Sensors* **2022**, *22*. doi:10.3390/s22186998.
40. Mead, C.A.; Mahowald, M. A silicon model of early visual processing. *Neural Networks* **1988**, *1*, 91–97. doi:https://doi.org/10.1016/0893-6080(88)90024-X.
41. Hanover, D.; Loquercio, A.; Bauersfeld, L.; Romero, A.; Penicka, R.; Song, Y.; Cioffi, G.; Kaufmann, E.; Scaramuzza, D. Autonomous Drone Racing: A Survey, 2023. doi:10.48550/ARXIV.2301.01755.
42. Ralph, N.O.; Marcireau, A.; Afshar, S.; Tothill, N.; van Schaik, A.; Cohen, G. Astrometric Calibration and Source Characterisation of the Latest Generation Neuromorphic Event-based Cameras for Space Imaging, 2022. doi:10.48550/ARXIV.2211.09939.
43. Salah, M.; Chehadah, M.; Humais, M.; Wahbah, M.; Ayyad, A.; Azzam, R.; Seneviratne, L.; Zweiri, Y. A Neuromorphic Vision-Based Measurement for Robust Relative Localization in Future Space Exploration Missions. *IEEE Transactions on Instrumentation and Measurement* **2022**, pp. 1–1. doi:10.1109/TIM.2022.3217513.
44. Ayyad, A.; Halwani, M.; Swart, D.; Muthusamy, R.; Almaskari, F.; Zweiri, Y. Neuromorphic vision based control for the precise positioning of robotic drilling systems. *Robotics and Computer-Integrated Manufacturing* **2023**, *79*, 102419. doi:https://doi.org/10.1016/j.rcim.2022.102419.
45. Muthusamy, R.; Ayyad, A.; Halwani, M.; Swart, D.; Gan, D.; Seneviratne, L.; Zweiri, Y. Neuromorphic Eye-in-Hand Visual Servoing. *IEEE Access* **2021**, *9*, 55853–55870. doi:10.1109/ACCESS.2021.3071261.
46. Hay, O.A.; Chehadeh, M.; Ayyad, A.; Wahbah, M.; Humais, M.A.; Boiko, I.; Seneviratne, L.; Zweiri, Y. Noise-Tolerant Identification and Tuning Approach Using Deep Neural Networks for Visual Servoing Applications. *IEEE Transactions on Robotics* **2023**, pp. 1–13. doi:10.1109/TRO.2023.3235586.
47. Rebecq, H.; Ranftl, R.; Koltun, V.; Scaramuzza, D. Events-to-Video: Bringing Modern Computer Vision to Event Cameras. *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)* **2019**.
48. Gallego, G.; Delbrück, T.; Orchard, G.; Bartolozzi, C.; Taba, B.; Censi, A.; Leutenegger, S.; Davison, A.J.; Conradt, J.; Daniilidis, K.; Scaramuzza, D. Event-Based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2022**, *44*, 154–180. doi:10.1109/TPAMI.2020.3008413.
49. Bing, Z.; Baumann, I.; Jiang, Z.; Huang, K.; Cai, C.; Knoll, A. Supervised Learning in SNN via Reward-Modulated Spike-Timing-Dependent Plasticity for a Target Reaching Vehicle. *Frontiers in Neurorobotics* **2019**, *13*. doi:10.3389/fnbot.2019.00018.
50. Schuman, C.D.; Kulkarni, S.R.; Parsa, M.; Mitchell, J.P.; Date, P.; Kay, B. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science* **2022**, *2*, 10–19. doi:10.1038/s43588-021-00184-y.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

51. Gehrig, D.; Loquercio, A.; Derpanis, K.G.; Scaramuzza, D. End-to-End Learning of Representations for Asynchronous Event-Based Data. Int. Conf. Comput. Vis. (ICCV), 2019.
52. Gehrig, M.; Scaramuzza, D. Recurrent Vision Transformers for Object Detection with Event Cameras, 2022. doi:10.48550/ARXIV.2212.05598.
53. Gehrig, M.; Millhäusler, M.; Gehrig, D.; Scaramuzza, D. E-RAFT: Dense Optical Flow from Event Cameras. International Conference on 3D Vision (3DV), 2021.
54. Barchid, S.; Mennesson, J.; Djéraba, C. Bina-Rep Event Frames: A Simple and Effective Representation for Event-Based Cameras. 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 3998–4002. doi:10.1109/ICIP46576.2022.9898061.
55. Bi, Y.; Chadha, A.; Abbas, A.; .; Bourtsoulatze, E.; Andreopoulos, Y. Graph-based Object Classification for Neuromorphic Vision Sensing. 2019 IEEE International Conference on Computer Vision (ICCV). IEEE, 2019.
56. Li, Y.; Zhou, H.; Yang, B.; Zhang, Y.; Cui, Z.; Bao, H.; Zhang, G. Graph-based Asynchronous Event Processing for Rapid Object Recognition. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 914–923. doi:10.1109/ICCV48922.2021.00097.
57. Fey, M.; Lenssen, J.E.; Weichert, F.; Müller, H. SplineCNN: Fast Geometric Deep Learning with Continuous B-Spline Kernels. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
58. Deng, Y.; Chen, H.; Xie, B.; Liu, H.; Li, Y. A Dynamic Graph CNN with Cross-Representation Distillation for Event-Based Recognition, 2023. doi:10.48550/ARXIV.2302.04177.
59. Alkendi, Y.; Azzam, R.; Ayyad, A.; Javed, S.; Seneviratne, L.; Zweiri, Y. Neuromorphic Camera Denoising Using Graph Neural Network-Driven Transformers. *IEEE Transactions on Neural Networks and Learning Systems* **2022**, pp. 1–15. doi:10.1109/TNNLS.2022.3201830.
60. Bronstein, M.M.; Bruna, J.; Cohen, T.; Veličković, P. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, 2021. doi:10.48550/ARXIV.2104.13478.
61. Schaefer, S.; Gehrig, D.; Scaramuzza, D. AEGNN: Asynchronous Event-based Graph Neural Networks. IEEE Conference on Computer Vision and Pattern Recognition, 2022.
62. You, J.; Du, T.; Leskovec, J. ROLAND: graph learning framework for dynamic graphs. Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 2358–2366.
63. Gehrig, D.; Scaramuzza, D. Pushing the Limits of Asynchronous Graph-based Object Detection with Event Cameras, 2022. doi:10.48550/ARXIV.2211.12324.
64. Gong, L.; Cheng, Q. Exploiting Edge Features for Graph Neural Networks. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 9203–9211. doi:10.1109/CVPR.2019.00943.
65. Wang, K.; Han, S.C.; Long, S.; Poon, J. ME-GCN: Multi-dimensional Edge-Embedded Graph Convolutional Networks for Semi-supervised Text Classification, 2022. doi:10.48550/ARXIV.2204.04618.
66. iniVation. DAVIS 346. https://inivation.com/wp-content/uploads/2019/08/DAVIS346.pdf.
67. Universal Robotics. USER MANUAL - UR10 CB-SERIES - SW3.15 - ENGLISH INTERNATIONAL (EN). https://www.universal-robots.com/download/manuals-cb-series/user/ur10/315/user-manual-ur10-cb-series-sw315-english-international-en/.
68. Guo, S.; Delbruck, T. Low Cost and Latency Event Camera Background Activity Denoising. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2023**, *45*, 785–795. doi:10.1109/TPAMI.2022.3152999.
69. Feng, Y.; Lv, H.; Liu, H.; Zhang, Y.; Xiao, Y.; Han, C. Event Density Based Denoising Method for Dynamic Vision Sensor. *Applied Sciences* **2020**, *10*. doi:10.3390/app10062024.
70. Delbrück, T. Frame-free dynamic digital vision. Proceedings of International Symposium on Secure-Life Electronics, Advanced Electronics for Quality Life and Society, Univ. of Tokyo, Mar. 6-7, 2008; Hotate, K., Ed.; nternational Symposium on Secure-Life Electronics, Advanced Electronics for Quality Life and Society 2008: Tokyo, 2008; pp. 21 – 26. nternational Symposium on Secure-Life Electronics, Advanced Electronics for Quality Life and Society 2008; Conference Date: March 6-7, 2008.
71. Fey, M.; Lenssen, J.E. Fast Graph Representation Learning with PyTorch Geometric. ICLR Workshop on Representation Learning on Graphs and Manifolds, 2019.
72. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 April 2023 doi:10.20944/preprints202304.0133.v1

73. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E.Z.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *CoRR* **2019**, *abs/1912.01703*, [1912.01703].
74. van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* **2008**, *9*, 2579–2605.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.