This paper has been accepted for publication at the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, Tucson (USA), 2025. ©IEEE

Combined Physics and Event Camera Simulator for Slip Detection

Thilo Reinold[1], Suman Ghosh[1] and Guillermo Gallego[1,2].
[1] Technische Universität Berlin, and Robotics Institute Germany.
[2] Einstein Center for Digital Future, and Science of Intelligence Excellence Cluster.

Abstract

Robot manipulation is a common task in fields like industrial manufacturing. Detecting when objects slip from a robot's grasp is crucial for safe and reliable operation. Event cameras, which register pixel-level brightness changes at high temporal resolution (called "events"), offer an elegant feature when mounted on a robot's end effector: since they only detect motion relative to their viewpoint, a properly grasped object produces no events, while a slipping object immediately triggers them. To research this feature, representative datasets are essential, both for analytic approaches and for training machine learning models. The majority of current research on slip detection with event-based data is done on real-world scenarios and manual data collection, as well as additional setups for data labeling. This can result in a significant increase in the time required for data collection, a lack of flexibility in scene setups, and a high level of complexity in the repetition of experiments. This paper presents a simulation pipeline for generating slip data using the described camera-gripper configuration in a robot arm, and demonstrates its effectiveness through initial data-driven experiments. The use of a simulator, once it is set up, has the potential to reduce the time spent on data collection, provide the ability to alter the setup at any time, simplify the process of repetition and the generation of arbitrarily large data sets. Two distinct datasets were created and validated through visual inspection and artificial neural networks (ANNs). Visual inspection confirmed photorealistic frame generation and accurate slip modeling, while three ANNs trained on this data achieved high validation accuracy and demonstrated good generalization capabilities on a separate test set, along with initial applicability to real-world data.
Project page: https://github.com/tub-rip/event_slip

1. Introduction

Pick-and-place tasks are a common application for robot manipulators, for example, in industrial manufacturing. Typically, these tasks involve a robot manipulator or arm equipped with a gripper and an object. The task is to grasp the given object with this gripper and move it to a designated position where it is released again. While the task involves several challenges such as determining grasp points, navigating obstacles, and implementing object detection, our work focuses on task observation as a crucial requirement for error handling. The observation can be performed using various sensors such as tactile sensors, cameras, lasers, or event cameras. The potential errors range from grasping the wrong object or failing to grasp entirely, to colliding with obstacles, deforming objects, or experiencing slips. A slip occurs when there is unwanted movement of the object while in the gripper's grasp. Specifically, a rotational slip involves movement around an axis, while a translational slip involves movement along an axis. Rotational slips commonly occur when the grasp point is not precisely aligned in the gravity direction with the object's center of mass. For example, when a box with an off-center mass distribution is grasped at its geometric center, it may rotate toward its center of mass if the friction between gripper and object is insufficient. While this rotation alone is problematic, any subsequent movement with the misaligned object can lead to further errors. Tasks like placing the rotated object or lifting it over obstacles may fail due to this

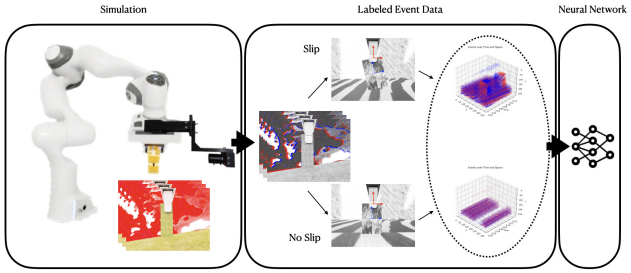


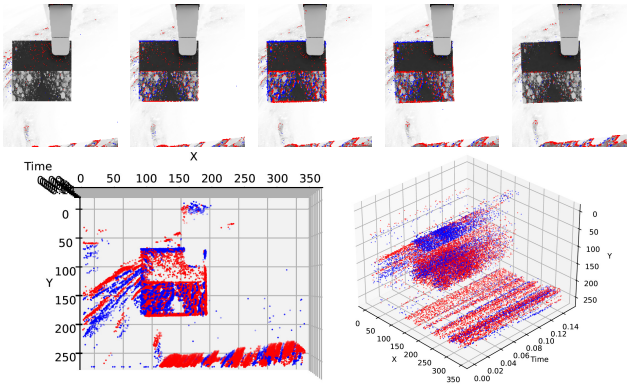
Figure 1. *Overview of the pipeline*: from physics simulation generating synthetic frames of a manipulation task with a Panda robot manipulator (left), through event generation and labeling of slip and non-slip cases (middle), to artificial neural network training (right). The simulation provides ground truth data through the robot's movements, which is used to label the generated event data for supervised learning of slip detection.

misalignment. In severe cases, particularly under high accelerations, the grasped object may completely detach or be ejected at high velocities, potentially causing damage.

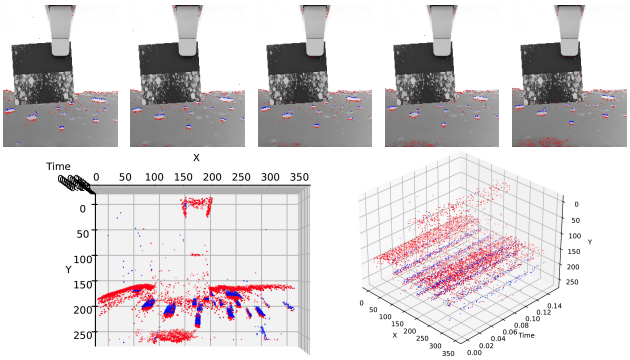
This work focuses on detecting slips in pick-and-place tasks using event cameras on a robot arm (Fig. 1). An event camera offers several beneficial features for this task [4, 11]: high temporal resolution, high dynamic range, and low power consumption. However, this paper emphasizes the camera's fundamental operating principle –generating events only when pixel brightness changes– as the key advantage for slip detection. Due to this property, events are only produced by movements relative to the camera, and when the camera is rigidly mounted to the gripper of the robot arm, only movements relative to the gripper generate events. Thus, an object held by the end effector will only produce events if it slips (see Fig. 2).

In the work of [1], a similar approach was explored using an event camera for slip detection. While their analytical methods successfully detected slips in specific scenarios, the challenge of parameter fine-tuning for generalization remained unsolved. This motivated us to explore data-driven approaches, which would mitigate parameter tuning. Upon reviewing related literature, we identified that collecting appropriate data represents one of the primary challenges in addressing this task. Having complete control over data quantity and complexity without excessive manual effort provides a significant advantage. This enables analytical approaches to start with simple setups and incrementally increase complexity, while also supporting data-driven methods through the generation of large, diverse datasets for both general and specific tasks. A simulator provides the means to achieve this control over data generation. It enables full control over scenarios (up to some approximation of real-world conditions), allowing for repeatable experiments with controlled variations and straightforward generation of ground truth data.

This paper presents a simulation pipeline that generates synthetic event data from photorealistic images, produced by physically accurate simulations of a basic pick-and-place task. To the best of our knowledge, it is the first time that such a simulation system is proposed. The system pipeline simulates a Franka Emika Panda robot manipulator, with a rigidly attached event camera, interacting with a box-like object. It allows modifications of object properties (shape, size, weight), textures, environmental lighting, and gripping positions to control dataset complexity. Leveraging the event-camera's principle of detecting only relative motion, our pipeline generates synthetic data specifically suited for slip detection (Fig. 1). To validate both this approach and the quality of the simulated data, we trained three artificial neural network (ANN) architectures (see Sec. 3.2.2) on the synthetic data and a real-world dataset, achieving assuring results on the slip detection task.



(a) **Slip**. Top: Frames and events vs. time. Bottom: 3D plots of events in space-time. Events produced by the object can be seen on the whole interval.



(b) **No slip**. Top: Frames and events vs. time. Bottom: 3D plots of events in space-time. No events appear on the object.

Figure 2. Visualization of two samples of the same simulation scenario, with or without slip.

2. Related Work

2.1. Robot Simulation

The use of simulators in robotics research has been well established; a notable example is the work of Shariq Iqbal *et al*. [9], who developed a photorealistic simulator with real-world physics to investigate *directional semantic grasping*. This task involves a robot grasping specific objects from predetermined directions within a shelf containing multiple items. The researchers successfully demonstrated sim-to-real transfer by training a deep reinforcement learning algorithm for closed-loop control of object grasping on simulated data and deploying it in real-world scenarios. While these results were promising, the authors acknowledged that their approach remained confined to specific use cases, and broader, more generalized solutions required further research. Moreover, their simulator was developed exclusively for conventional (*i.e*., frame-based) vision sensors and has not been made available to the research community.

Commercial and open-source physics simulators, such as

NVIDIA's Isaac Sim [13] and PyBullet [2], provide sophisticated simulation environments. These platforms use game engines like *Bullet* and *PhysX* to deliver realistic physics simulations and photorealistic rendering capabilities.

PyBullet and Isaac Sim support robotic simulations, synthetic data generation, and various machine learning applications through their Python interfaces, but they miss event-based data. Also, to use these simulators for slip detection, the scenario must first be set up by configuring the environment, defining object properties, integrating a robotic manipulator, and implementing the required sensors.

2.2. Event-Based Slip Detection

Recent years have seen numerous publications in the field of event-based cameras, with several focusing on manipulation tasks [4]. However, only a limited number of these works address in-hand tracking or slip detection.

Taunyazov *et al*. [15] implemented a multi-modal approach combining two event-based sensors: a NeoTouch tactile sensor and an event camera. Their Visual-Tactile Sensing system (VT-SNN) integrates data from both sensors to train a neural network for two distinct tasks: object detection, which distinguishes between visually identical objects of different weights, and rotational slip detection on a single object. While their results demonstrate the potential of this approach, the evaluation was limited by a small dataset, lack of object diversity, and brief testing intervals.

Niklas Funk *et al*. [3] introduced an event-driven approach using their optical tactile sensor, *Evetac*. The sensor consists of a soft, non-transparent silicone gel with embedded markers attached to the gripper, where deformations from object contact and movement are captured by an event camera positioned behind the gel. While similar optical tactile sensors exist, they traditionally use frame-based cameras. The system achieved impressive results with slip detection at 1kHz, albeit in a constrained scenario with limited object variety and no manipulator movement. The sensor's independence from environmental motion and lighting conditions significantly reduced the dataset complexity requirements. The work stands out for its reproducibility through commercial components, 3D-printable parts, and open-source algorithms, though it requires time-consuming manual grasp execution during data collection and gel preparation for ground-truth sessions.

Rajkumar Muthusamy *et al*. [12] developed an event-based finger vision system for slip detection and suppression during manipulation tasks. Their system employs a parallel gripper with translucent finger plates and an event camera. While achieving a high slip detection rate of 2kHz, the system faces several constraints: a highly restricted testing environment, limited object movements, dependence on sufficient object surface texture, and the requirement for manual data collection.

In [1], various analytical approaches to rotational slip detection are investigated with a setup similar to the one we consider, by using the event rate, event-based optical flow [14, 17], etc. The methods exploit the principle that objects generate events only during relative motion between the camera and gripper. Compared to previous approaches, this work examines longer manipulation periods with more dynamic movements. However, the camera viewpoint presents a limitation, as rotational slip movements toward or away from the camera are difficult to detect. Observing from a lateral perspective could improve the detection of such movements. The authors concluded that more diverse scenarios were needed for robust and generalizable results.

The reviewed works demonstrate promising approaches to event-based slip detection but share common limitations in their evaluation. A recurring challenge is the time-consuming nature of data collection, with most approaches requiring manual execution of grasps and complex setup procedures. Additionally, the existing studies are typically restricted to controlled environments with limited object variety and constrained movements. While [1] examines longer manipulation periods, all approaches would benefit from more diverse scenarios to achieve robust and generalizable results. These limitations highlight the need for a systematic approach to generate comprehensive datasets for event-based slip detection. A simulation environment could address these challenges through automated data generation in various scenarios, objects, and grasp conditions.

3. Simulation and Slip Detection Pipeline

We present a complete pipeline that covers both data generation and data-driven slip detection (through ANN training), as illustrated in Fig. 3. The pipeline consists of two independent modules: a data generation system and an ANN training framework. These modules can be utilized either as an integrated pipeline or as standalone components.

3.1. Dataset Generation

The complete pipeline for the generation of event-based simulated datasets consists of four steps, which we describe in the upcoming sections. The first step is the parameter generation for each dataset. The second step is the physics simulation, which generates frames and position values of the object and the gripper. The third step is the simulation of event camera data. The final step is the calculation of the angular difference for every generated sample, which serves as ground-truth (GT) data.

3.1.1 Parameter Generation

The input for this step determines the size of the dataset, whether the dataset is split into training and test sets, and

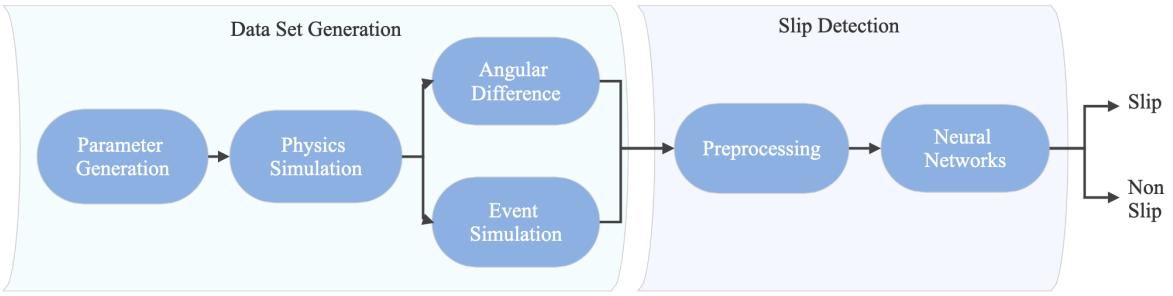


Figure 3. Data simulation and slip detection pipeline.

how the parameter values of the simulation scene are selected. A dataset consists of individual samples, where each sample contains all the data generated from a simulation. The size of the datasets has no constraints but the available computation time is usually the limiting factor. A split into training and test sets guarantees that no parameter value of the test set is previously used in the trainings set. The parameters for a simulation are either generated by an exhaustive combination of a list of selected parameters or by random selection from a value range (for a detailed description of the parameters see Sec. 3.1.2).

3.1.2 Physics Simulation

This step executes the physics simulation to generate gripper and object orientations for ground truth, along with synthetic images (*i.e.*, frames) for event data generation. The simulation utilizes NVIDIA's Isaac Sim simulator [13], a powerful robotics platform that provides realistic physics and produces photorealistic simulations. While Isaac Sim offers a graphical editor for manual scenario building and testing, we implement programmatic control through Python scripts. Our approach enables access to all configurable parameters as both inputs and outputs, making it highly versatile for data generation.

The basic scene consists of a Franka Emika Panda robot manipulator mounted on a flat surface, a camera focused on the gripper, and a simple cuboid for pick and place operations. A light source is positioned above the scene, and the entire setup is enclosed within a sphere to control background texture and lighting (see Fig. 4). The simulation setup offers various adjustable parameters whose values are generated in Sec. 3.1.1. These include the textures of the cuboid, ground surface, and the inside of the surrounding sphere, as well as lighting conditions through the sphere's emission and the internal light source's brightness and position. The gripping points on the cuboid, the cuboid's width, height, and mass are adjustable.

rrences can be influenced through several parameters like the object's mass, width, and gripping position. For example, gripping positions far from the center increase slip probability. While there is no direct parameter to toggle slip instances, they can be effectively controlled via parameter combinations. Maximum mass settings guarantee slip occurrence, while minimum mass prevents it. Intermediate values produce varying results depending on other parameters, *e.g.*, a fixed-mass cuboid might slip when gripped at its edge but remain stable when gripped centrally. This approach allows most parameters to generate both slip and no-slip cases while enabling various slip types at different speeds and positions along the trajectory.

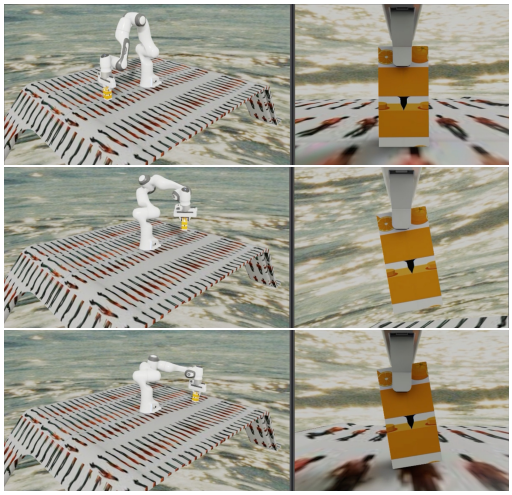


Figure 4. Simulator Scene. The left side shows the overall scene, and the right side shows the view from the camera attached to the robot arm. Execution of a whole pick-and-place task is shown sequentially from top to bottom.

3.1.3 Event Generation

Event generation is performed using the v2e simulator [7], which converts frames into synthetic event-camera data after interpolating between frames using a pre-trained ANN. Since the maximum rendering frame rate of the Isaac Sim simulation was limited to 60Hz, an upsampling mechanism was needed to achieve realistic event generation. The simulated frames, being free of motion blur, are particularly

well fit for this upsampling and event generation process. Through comprehensive noise modeling, the simulator generates events that closely approximate real event camera data. The frames from the Isaac Sim physics simulation serve as input to v2e, along with several configurable parameters. The key parameters in this work are the event threshold (contrast sensitivity of the event generation model [4, 5]), input frame rate (60Hz), and output resolution of 346× 260 pixels, which matches the specifications of the DAVIS346 camera [8]. While additional v2e parameters can be adjusted as needed, these were identified as the most critical for our application. The simulation produces two output files: an event data file where each row contains an event's timestamp, x-coordinate, y-coordinate, and polarity, and a parameter file documenting the v2e configuration used in the simulation.

3.1.4 Angular Difference

To establish ground truth, we calculate the angular distance between the object and gripper (see Fig. 5). During each simulation step, the positions and orientations of the gripper are computed and recorded. The initial positions of both the gripper and cube are stored as rotation matrices B_{gripper} and B_{cube}. These matrices serve as reference points for calculating subsequent angular differences, enabling measurement of relative changes from the initial configuration. The angular difference θ (in radians) is calculated at each iteration using the following formulas:

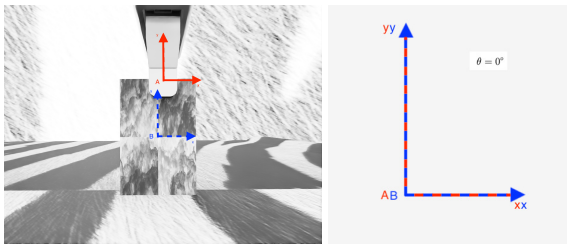
$$\begin{aligned} \Delta M_{\text{gripper}} &= M_{\text{gripper}} \cdot B_{\text{gripper}}^{\top} \\ \Delta M_{\text{cube}} &= M_{\text{cube}} \cdot B_{\text{cube}}^{\top} \\ \theta &= \left| \arccos\left(\frac{\text{tr}(\Delta M_{\text{gripper}}^{\top} \cdot \Delta M_{\text{cube}}) - 1}{2} \right) \right| \end{aligned} \tag{1}$$

where M_{gripper} and M_{cube} represent the rotation matrices that describe the current orientation of the gripper and cube, respectively. The classification of samples as slip or non-slip requires careful consideration of angular differences θ. In this work, two thresholds were defined: θ_{slip} for the minimum angular difference indicating slip, and $\theta_{\text{non-slip}}$ for the maximum allowed angular difference in non-slip cases. Defining these thresholds is challenging as angular changes can occur even without visible slippage, due to factors such as camera resolution limitations, extremely slow movements, or camera perspective. These subtle changes might not be perceptible to the human eye, yet they still register as measurable angular differences.

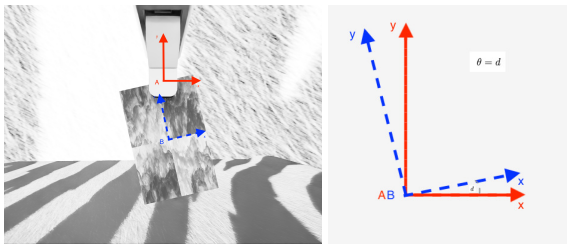
3.2. Slip Detection

3.2.1 Preprocessing

The preprocessing serves four key objectives: (i) breaking down each sample's event stream into fixed-length subsamples, (ii) generating ground truth labels by categorizing subsamples as slip or non-slip, (iii) optionally visualizing the data to ensure human interpretability, and (iv) formatting the data to fit the ANN slip classifier. The subsample length is fixed at 0.16 seconds in this work, balancing rapid slip detection (comparable to VT-SNN's 0.15 seconds) with practical visualization considerations. The physics simulator generates frames at 60 Hz (every 0.016 seconds in simulation time), resulting in exactly 10 frames per subsample, streamlining the visualization process.



(a) Start of the sequence, prior to the slip. The orientations of the two coordinate systems agree.

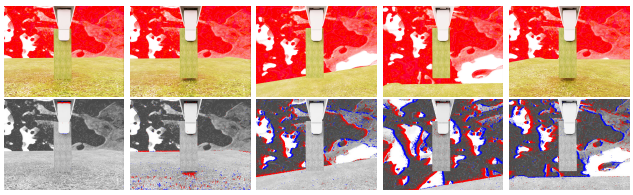


(b) Half a second into the sequence, a rotational slip has occurred. The orientations of the two coordinate systems are no longer aligned, as shown by their angular difference.

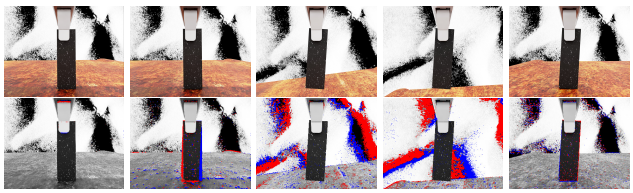
Figure 5. Two images of a slip sequence, with their coordinate systems marked in red and blue, illustrate a rotational slip and the resulting angular difference.

The ground truth labeling algorithm requires two angular threshold parameters: a slip threshold θ_{slip} that defines the minimum angular difference to classify a subsample as slip, and a non-slip threshold $\theta_{\text{non-slip}}$ that defines the maximum angular difference to classify a subsample as non-slip (see Sec. 3.1.4). When these thresholds are set to the same value, the dataset is cleanly divided into slip and non-slip categories. Using different values creates an exclusion zone: subsamples with angular differences falling between the thresholds are omitted from the dataset. This exclusion can lead to clearer decision boundaries during ANN slip-detection training.

For each subsample, videos and individual frames, both with and without events, are created for visualization purposes. The visualization can be deactivated through input settings to save computational time and disk space. To match the network architectures of [15], spatial data is cropped from 346×260 px to 200×250 px, which also helps focus on the region of interest by reducing background



(a) Non-slip sample.



(b) Slip sample.

Figure 6. Visualization of two pick-and-place sequences across five key movement phases (from left to right): (1) initial position, (2) object lifting, (3) lateral tilt, (4) back to vertical gripper orientation, and (5) final placement. Top row: color frames from the physics simulator. Bottom row: synthesized event data overlaid on grayscale frames.

events. For each subsample, a .mat file is generated, ready for use with ANNs for slip detection. These files contain the events and are named with the label - rotation.mat or stable.mat. To ensure balanced data, slip and non-slip subsamples are counted separately, and random samples from the larger subset are removed until both categories contain equal number of subsamples.

3.2.2 Artificial Neural Networks

The implementation follows three artificial neural network (ANN) architectures, as configured in [15]. Each dataset is divided into training and validation sets using a 80:20 split ratio. The subsamples, each 0.16 seconds in duration, are processed differently according to the network architecture: 150 time bins for the Spiking Neural Network (SNN) and Multi-Layer Perceptron (MLP) architectures, and 350 time bins for the 3D-CNN (Convolutional Neural Network) architecture. The ANNs perform binary classification between slip and non-slip subsamples. Two optimizers are employed to train the ANNs, as detailed in the experiments.

4. Slip Detection Datasets

Three datasets were created to validate the data generation pipeline and conduct slip detection experiments: two simulated sets (*Simple Set*, *Complex Set*) and a real-world data set (*Real Set*). The *Simple Set* consists of 192 simulations using only two texture combinations and little diversity. The *Complex Set* contains 1200 simulations incorporating 48 different textures and high diversity, while maintaining the specified parameter constraints. This set is split into training and testing subsets. These distinct sets serve two purposes: demonstrating the ability to control dataset scale and complexity, and providing varying levels of difficulty for slip detection (*i.e.*, classification) experiments.

Property	**Simple Set**	**Complex Set Training**	**Complex Set Test**
Width of cuboid	2	1000	200
Height of cuboid	2	1000	200
Mass of cuboid	2	1000	200
Texture sets	2	1000	200
Light sphere position	2	1000	200
Light sphere brightness	1	1000	200
Background sphere brightness	1	1000	200
Gripping position horizontally	3	1000	200
Gripping position vertically	2	1000	200
Total	192	1000	200

Table 1. Selected property values across sets. The *Simple Sets* was created by combining the extrema of the selected properties and a continuous background lighting. For both subsets of the *Complex Set*, every parameter value was selected randomly for every simulation, with a strict separation of values between both.

Each simulation produces a sample containing approximately 1 million events, 450 colored frames at 1730×1300 pixel resolution, and 2×450 quaternions describing gripper and object orientations, totaling nearly one gigabyte (GB). While each pick-and-place task simulation represents 7-8 seconds of virtual time, generating a single sample requires about 6 minutes. However, the simulator's parallel processing capabilities allow for simultaneous execution of multiple simulations. For the generation of these datasets, 8 simulations were run in parallel, resulting in an average computation time of 2.7 minutes per sample on a system with two NVIDIA RTX A6000 GPUs and a dual AMD EPYC 7543 processor featuring 128 logical cores.

4.1. Simple Set

The *Simple Set* consists of 192 samples without an explicit test set and totals 185 GB. The data's simplicity is achieved by limiting parameter variations: only two texture combinations for the cuboid, surface, and background, two positions for the sphere light, and extreme values for all other parameters. These extremes include the highest and lowest cuboid masses (to guarantee slip and non-slip cases), maximum and minimum heights, maximum and minimum widths, and gripping positions at the leftmost, middle, and rightmost points, both at the highest and lowest possible positions. All these parameter options were exhaustively combined to generate the samples (see Tab. 1). The event simulation parameters, including the contrast sensitivity and upsampling rate, were set to default values to match a DAVIS346 camera [8] at an input frame rate of 60 Hz.

4.2. Complex Set

The *Complex Set* (Fig. 7) consists of a training set of 1000 samples and a test set of 200 samples. The training

set totals 883 GB, though this size is primarily due to visualization data (images and videos). Excluding these visualization files (Fig. 6) and retaining only the event files and ground-truth data reduces the storage to approximately 87 GB. Data diversity was achieved through multiple parameter variations. A pool of 48 unique textures was split with an 80:20 ratio between training and test sets, with each sample randomly receiving three textures for the surface, cuboid, and background. Illumination varied through two random components: the enclosing sphere's brightness ranged from very bright to dim, while a light sphere was randomly positioned on a hemisphere above the scene, with its intensity varying from bright (casting sharp shadows) to dim (producing minimal shadows) (see Tab. 1). The cuboid's parameters covered shape configurations from wide and flat to tall and thin, with masses ranging from guaranteed slip to guaranteed non-slip conditions. The gripper's position was randomized between the object's edges and middle, and from maximum grip depth to the top edge. While parameter values can occur multiple times within either the training or test set, strict separation between sets was maintained –no exact value appears in both. However, due to the continuous nature of the parameters, values can be similar across sets.

The event simulation parameters, including the contrast sensitivity and upsampling rate, were configured to match a DAVIS346 camera [8] at an input frame rate of 60 Hz.

4.3. Real Set

The *Real Set* (Fig. 8) comprises five samples, including three slipping sequences and two non-slipping ones. The ground-truth labels were assigned through manual observation of the combined frame and event data. The experimental setup closely follows [1], using a Panda robot mounted on a flat white surface with a DAVIS346 camera attached to the gripper. The primary modification is a custom 3D-printed camera mount that provides a side view of the gripper and object, replacing the previous front view configuration. This physical setup served as the reference model for the simulator development. The manipulation object chosen was a moderately textured book. While slightly larger than its simulated counterpart and showing minor deformation during grasping, it is similar to the simulated data.

5. Experiments

To evaluate the usefulness and validity of the generated datasets, experiments were performed by training different ANN slip-detection architectures. The chosen architectures are inspired by the experiments in [15] (Sec. 3.2.2). The networks include an MLP, an SNN, and a 3-dimensional CNN (3D-CNN). All three architectures were trained on both sets (*Simple Set* and *Complex Set*), separately, with varying parameters. The data was preprocessed and parameter sweeps were performed using [16], with lists of possible values provided for each parameter. For every run, parameter values were selected randomly. The parameters included batch size, learning rate, and two optimization algorithms: RMSProp [6] and Adam [10].

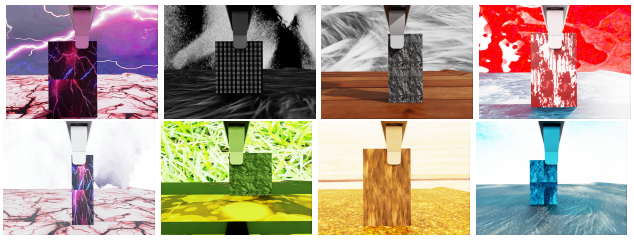


Figure 7. Frames of the *Complex Set* with varying parameters.

Figure 8. Left: Experimental setup for real-world data collection. Right: Event data and frames captured during a slip sequence from real-world experiments.

5.1. Preprocessing

To prepare the data for training and testing, the samples of the *Simple Set*, *Complex Set* and the *Real Set* were preprocessed (Sec. 3.2.1). The preprocessing involves splitting the samples into subsamples of 16 ms. These subsamples were then segmented differently for each network architecture: 150 bins of about 1 ms for the SNN and MLP, and 350 bins of about 0.46 ms for the 3D-CNN. We labeled the subsamples based on their angular difference change using two thresholds (see Sec. 3.1.4): subsamples showing changes greater than $1.0°$ are labeled slip, whereas those with changes less than $0.1°$ are non-slip. Subsamples with values between these thresholds were considered uncertain and discarded. To balance the datasets, the larger set of labeled subsamples was reduced by random removal until it matched the size of the smaller set. Further preprocessing steps included cropping the events to 200×250 px and generating visualizations. Finally, the resulting subsample sets were randomly split into training and validation sets using an 80:20 ratio. This resulted in 918 training and 230 validation subsamples for the *Simple Set*, 11554 training, 2888 validation, and 2682 test subsamples for the *Complex Set* and 81 subsamples for the *Real Set*.

5.2. Slip Detection on Simulated Data

The results for both the *Simple Set* and *Complex Set* are presented in Tab. 2. The following section focuses on the *Complex Set* results, while a detailed description of the *Simple Set* is provided in the Supplementary Material.

We also evaluated on the *Complex Set* using the same three ANNs (MLP, SNN, and 3D-CNN), comprising 225

Architecture	Trained on	Validation Accuracy	Test Accuracy	Real Data Accuracy
MLP	*Simple Set*	98%	N/A	N/A
SNN	*Simple Set*	96%	N/A	N/A
3D-CNN	*Simple Set*	99%	N/A	N/A
MLP	*Complex Set*	72%	69%	63%
SNN	*Complex Set*	70%	69%	59%
3D-CNN	*Complex Set*	80%	78%	63%

Table 2. Data-driven slip detection performance across datasets. Classification accuracies evaluated on the *Simple Set* (validation only), the *Complex Set* (validation and test), and the *Real Set* (pre-trained on *Complex Set*).

training runs in total. They achieved lower accuracies than on the Simple Set, with the 3D-CNN achieving the highest validation accuracy at 80%, while both SNN and MLP performed at approximately 70%. Importantly, all three architectures demonstrated consistent performance when evaluated on the test set, with test accuracies closely matching their respective validation accuracies.

For the MLP architecture, the parameter sweep yielded its best performance with a validation accuracy of 72% and a cross entropy loss of 0.001. This was achieved using the Adam optimizer with a learning rate of 0.0005 and a batch size of 512. Most validation accuracies ranged between 62% and 72%, with a few runs resulting in 50% accuracy. The experiments revealed that lower learning rates (0.005 and below) consistently outperformed higher values, while batch sizes between 128 and 1024 yielded superior accuracies compared to smaller batch sizes below 128. Final evaluation on the *Complex Set* test set demonstrated an accuracy of 69% with a corresponding loss of 0.009.

For the SNN architecture, the parameter sweep yielded its best performance with a validation accuracy of 70% and a spike loss of 1175. This was achieved using the RMSProp optimizer with a learning rate of 0.00005 and a batch size of 512. Most runs achieved validation accuracies between 67% and 69%, with a few runs resulting in accuracies below 55%. The higher performing configurations consistently used learning rates between 0.00005 and 0.006, and generally performed better with batch sizes above 256. Final evaluation on the *Complex Set* test set demonstrated an accuracy of 69% with a corresponding loss of 1262.

For the 3D-CNN, the parameter sweep achieved its best performance with 80% accuracy and 0.026 cross entropy loss, using the RMSProp optimizer, a learning rate of 0.0001, and a batch size of 16. High-performing runs ranging from 75% to 80% accuracy, moderate runs around 70%, and poor performers between 50% to 60%. Consistent with previous findings, good performance correlated with smaller batch sizes (16 to 256) and learning rates (0.0001 to 0.002). The trained network was also evaluated on the test set, achieving an accuracy of 78% and a loss of 0.01.

5.3. Slip Detection on Real Set

The three best-performing ANNs from Section 5.2 were evaluated on the *Real Set*. The networks achieved accuracies of 63% (MLP, cross entropy loss: 0.06), 59% (SNN, spike loss: 1336), and 63% (3DCNN, cross entropy loss: 0.8) on this dataset (see Tab. 2), which indicates the models' capability to process real-world data.

6. Conclusion

This work presents a simulator for studying slip detection with event cameras in pick-and-place tasks, leveraging the event-camera's ability to isolate relative motion. It supports creating large datasets with configurable complexity, enabling both analytical and data-driven approaches. It features photorealistic rendering, physics-based simulation, synthetic event generation, and rich visualization for dataset analysis and edge-case investigation.

Two simulated datasets were created and used to validate the simulator's effectiveness for slip detection applications through both qualitative visual analysis and machine learning evaluation. The visual analysis involved assessing the photorealistic quality of images, the realism of object movements, and the ability of the human eye to differentiate between slips and non-slips in the images and the visualized events. The results were positive: the images exhibited a high degree of photorealism, object movements appeared natural, and most slip events were visually detectable.

All three ANNs (MLP, SNN and 3D-CNN) achieved validation accuracy rates above 95% on the *Simple Set*. While training on the *Complex Set* yielded lower accuracies, the performance on a never-before-seen test set containing new textures and scenarios remained consistent with the validation results. The 3D-CNN demonstrated the strongest performance, achieving 80% validation accuracy and 78% accuracy on the test set. These consistent results between validation and test sets suggest the models generalized well to new data rather than overfitting to the training set. Initial evaluation on a small real-world dataset showed accuracies around 60%, indicating the potential for slip detection transfer from simulation to real-world applications.

Acknowledgment

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2002/1 "Science of Intelligence" – project number 390523135. Funded by Amazon Research Award. We thank Profs. Toussaint and Raisch for support with the robotic manipulator and its software.

Supplementary Material

Slip Detection on Simple Set

A total of 495 training runs were executed over all three neural networks (MLP, SNN, and 3D-CNN). Each ANN achieved validation accuracy exceeding 95% with optimal parameter settings, with the 3D-CNN reaching 99.59% accuracy. The results are reported in Tab. 2.

The MLP parameter sweep achieved its best performance with 98% validation accuracy and 0.003 cross-entropy loss, using the RMSProp optimizer, a learning rate of 0.003, and a batch size of 128. Accuracies showed distinct clustering around either 90% or 50%. Poor results tended to occur with combinations of smaller batch sizes and higher learning rates. RMSProp and Adam optimizers showed comparable performance. For high-accuracy runs, loss values ranged from 0.0007 to 0.1.

The SNN's best configuration achieved 96% validation accuracy with a spike loss of 178, using the RMSProp optimizer, a learning rate of 0.011, and a batch size of 512. Training showed robust performance with accuracies consistently near or above 90% across all parameter combinations. Loss values varied widely, with some runs around 280 and others below 60 or even 20.

The 3D-CNN achieved its peak performance with 99.59% validation accuracy and 0.0035 cross-entropy loss, using the RMSProp optimizer, a learning rate of 0.002, and a batch size of 16. Accuracies clustered either above 80-90% or around 50%. High accuracies were predominantly achieved with small batch sizes (8 to 32) and learning rates below 0.020. Loss values stayed under 0.045, with high-accuracy runs typically below 0.01.

References

[1] Albert Bhagwan Baharuni. Exploration of methods for in-hand slip detection with an event-based camera during pick-and place motions. Master's thesis, TU Berlin, 2021.

[2] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2021.

[3] Niklas Funk, Erik Helmut, Georgia Chalvatzaki, Roberto Calandra, and Jan Peters. Evetac: An event-based optical tactile sensor for robotic manipulation. *IEEE Trans. Robot.*, 40:3812–3832, 2024.

[4] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jörg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(1):154–180, 2022.

[5] Guillermo Gallego, Christian Forster, Elias Mueggler, and Davide Scaramuzza. Event-based camera pose tracking using a generative event model. arXiv:1510.01972, 2015.

[6] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[7] Yuhuang Hu, Shih-Chii Liu, and Tobi Delbruck. v2e: From video frames to realistic DVS events. In *IEEE Conf. Comput. Vis. Pattern Recog. Workshops (CVPRW)*, pages 1312–1321, 2021.

[8] iniVation AG. https://inivation.com/wp-content/uploads/2019/08/DAVIS346.pdf. [Accessed October 25, 2024].

[9] Shariq Iqbal, Jonathan Tremblay, Thang To, Jia Cheng, Erik Leitch, Andy Campbell, Kirby Leung, Duncan McKay, and Stan Birchfield. Toward sim-to-real directional semantic grasping. *arXiv e-prints*, 2020. 1909.02075.

[10] Diederik P. Kingma and Jimmy L. Ba. Adam: A method for stochastic optimization. *Int. Conf. Learn. Representations (ICLR)*, 2015.

[11] Patrick Lichtsteiner, Christoph Posch, and Tobi Delbruck. A 128×128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE J. Solid-State Circuits*, 43(2):566–576, 2008.

[12] Rajkumar Muthusamy, Xiaoqian Huang, Yahya Zweiri, Lakmal Seneviratne, and Dongming Gan. Neuromorphic event-based slip detection and suppression in robotic grasping and manipulation. *IEEE Access*, 8:153364–153384, 2020.

[13] NVIDIA Corporation . https://developer.nvidia.com/isaac/sim. [Accessed October 25, 2024].

[14] Shintaro Shiba, Yoshimitsu Aoki, and Guillermo Gallego. Secrets of event-based optical flow. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 628–645, 2022.

[15] Tasbolat Taunyazov, Weicong Sng, Hian-Hian See, Brian Lim, Jethro Kuan, Abdul Fatir Ansari, Benjamin C. K. Tee, and Harold Soh. Event-driven visual-tactile sensing and learning for robots. *arXiv e-prints*, 2020. 2009.07083.

[16] Weights & Biases. Weights & biases: Machine learning experiment tracking, 2024. Accessed November 25, 2024.

[17] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. EV-FlowNet: Self-supervised optical flow estimation for event-based cameras. In *Robotics: Science and Systems (RSS)*, pages 1–9, 2018.