Over the past 60 years, systems that analyze and interpret camera images have become an integral part of our lives. Our cars observe their surroundings and identify obstacles while monitoring driver behavior and condition, our phones unlock based on facial recognition, and self-checkout machines independently recognize selected products. However, computer vision still struggles with challenges in specific circumstances. Imagine a drone moving rapidly among trees at sunset tasked with recognizing objects observed by its mounted camera. When reviewing the recorded images several issues become apparent. First, the observed scene will be blurred - a common issue at high speeds. Additionally, light filtering through the leaves will significantly impact image clarity - the set exposure time prevents proper recording of both bright and dark parts of the scene. What's more, if the drone uses vision system data for navigation (e.g., obstacle avoidance), necessary operations must be performed in real-time with minimal delays. Finally, the system must be energy efficient since it relies solely on an onboard battery.

This scenario motivates the proposed research, aiming to develop energy-efficient, real-time, low-latency, and accurate object detection systems that use modern event cameras inspired by the human eye and perform computations on specialized hardware platforms - FPGA programmable logic. Traditional cameras, known as "frame-based" synchronously record brightness values for the entire sensor matrix at set intervals (known as FPS - frames per second). We receive information about the observed scene only when a "video frame" is recorded. In contrast, event cameras record only changes in brightness (increase or decrease), capturing only changes in the scene - movement or brightness changes. This approach results in lower average energy consumption and excludes redundant information. Moreover, each pixel operates independently - instead of the familiar "photos", the data from this sensor forms a point cloud, with each point defined by pixel coordinates, precise time of brightness change, and polarity, indicating the direction of the change. This eliminates the common exposure time issue, making the camera effective in varying lighting conditions. Event cameras can generate over a billion events per second, preventing motion blur.

The challenge remains to process this unique, non-traditional frame data in real-time. The problem lies in that the most effective detectors use computationally intensive deep neural networks. The proposed project will research methods to adapt event data and neural networks for specialized computational platforms such as FPGAs, which are notable for low energy consumption (thus efficiency) and the ability to perform many computations simultaneously. Like event cameras, reconfigurable systems differ significantly from other platforms. During the research, their characteristics (e.g., limited memory resources, complexity in executing sophisticated calculations) must be considered at the algorithm design stage. The research will apply methods for adapting event vision systems to FPGAs, such as quantization (reducing the precision of used values), optimizing memory usage and typical neural network calculations, and employing hardware-friendly architectures (e.g., graph neural networks). It will also explore the possibility of appropriately dividing operations between different computational platforms - in heterogeneous systems, some tasks can be handled by a standard processor, while others by a programmable logic. A key element of FPGA logic programming is the verification of the hardware modules - thorough testing of the system through simulation and evaluation on the hardware platform.

The project's end result will be an event-based object detector fully implemented on a heterogeneous SoC FPGA platform, meeting the specified requirements. The prepared code will be published in an open repository, and the results will be presented as a conference paper or in peer-reviewed journals (preferably in open access, if possible).