Create an object detection runtime application

Whitepaper

KEBA
Automation by innovation.

Introduction

In this white paper we will look at how to create an AI vision application on a KEBA controller **using our new industrial AI extension module: the AE 550**.

Creating an AI vision application consists of two major steps. Firstly, we need to train the model so that it can recognize the right objects we want to detect in the input image. **We have already written a whitepaper which describes this step**. Secondly, we need to create a runtime application that executes the trained model and then interprets and uses the results correctly in our PLC application. While basic knowledge of AI is required for training the model, the creation of the runtime application is pure programming task. Due to the deep integration of the AI stack into the KEBA controller, no special knowledge regarding AI is required to create such an application.

The open KEBA control platform offers two options for creating such an application. On the one hand, **the application can be developed directly in the IEC code i.e. in KeStudio**. Here, special function blocks are provided to make it as simple as possible. However, there is also the option of developing the application in C/C++ or Python and communicating directly with the AI accelerator via an API.

AE 550

The AE 550 is a high-performance, compact hardware inference accelerator module that can be connected directly to a KEBA controller from the C5 series as an extension module. It is designed to meet the demanding needs of modern industrial applications. It features an open architecture that allows a highly flexible integration and customization. With real-time capability, it ensures immediate processing and response, which is crucial for time-sensitive operations. Data sovereignty and security are prioritized, ensuring that all data is handled with the highest standards of protection.

AE 550

Figure 1 – KeControl CP520/C extended by the AI Extension Module AE 550 (plugged in on the left-hand side)

The big difference to other industrial AI accelerators in industrial PCs or controllers is the seamless integration into the KEBA control system. This allows AI tasks to be performed directly from the PLC code. Built for industrial long-term availability, this accelerator is both durable and reliable, making it a sustainable choice for long-term projects. It is also optimized for energy and cost efficiency, reducing operational expenses while maintaining high performance.

The advantages of the deep integration of the AI and vision system can be described very well using the example of a modern robot packaging solution. In such an application, up to three different controllers and industrial PCs are installed with current solutions available on the market. First of all, a robot controller is needed to control the picking robots. Then it needs a camera system or an industrial PC that takes over the vision tasks, possibly carrying out a visual quality inspection and providing the robot with the coordinates of the objects to be picked. Finally, a PLC is required as well to connect all the other sensors in order to form the user and control the application process interface.

All these systems must be parameterized and/or programmed. This often requires completely different tools from different manufacturers, because most of the time the vision system provider does not provide a robot control nor PLC or vice versa. There are hardly any specialists who can successfully handle all the systems. This means that different people are often needed to commission or maintain a machine application.

With the **KEBA Kemro X industrial automation platform**, only a single control system that bundles the PLC, robotics and all AI functionalities is used. Likewise, only a single tool is needed to program all these system parts. This makes it easier for users to operate and for application engineers to program the system. Moreover: it also reduces the time needed to train new employees.

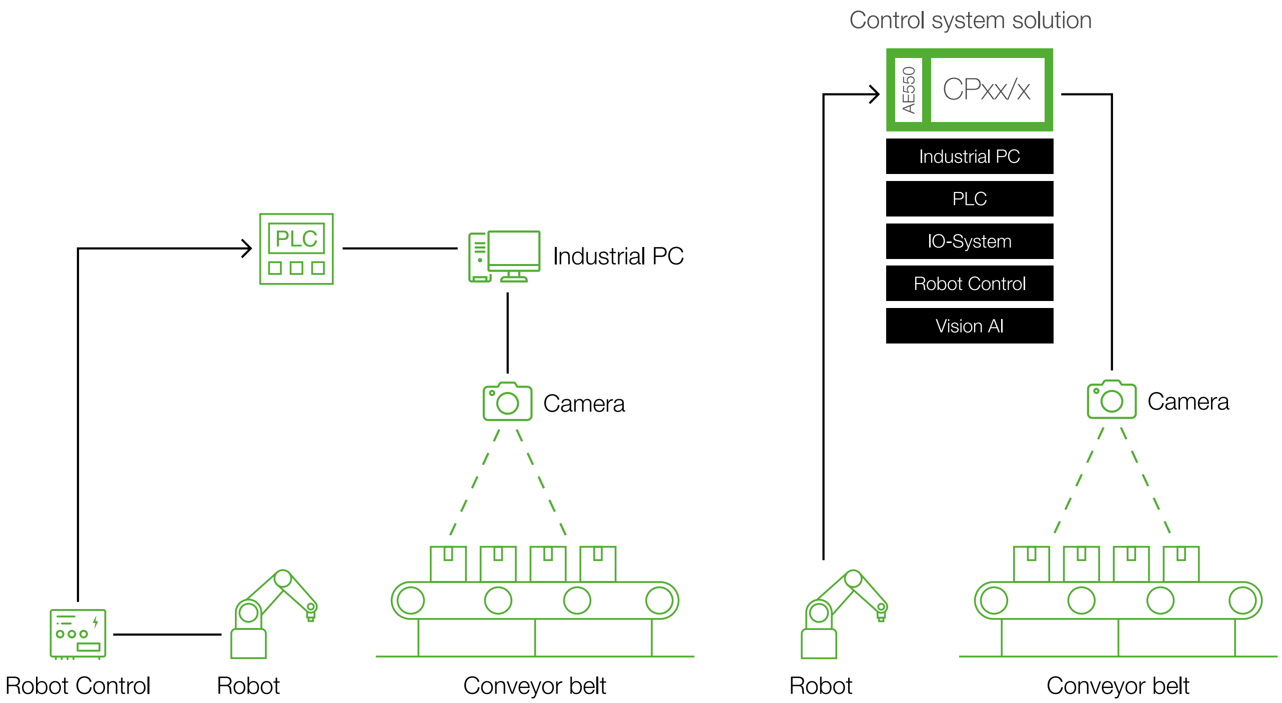

Figure 2 - Comparison of a conventional vision & control solution with the Kemro X solution

Processing steps in an AI runtime Application

Now we know the advantages of the KEBA AI extension module, let‘s take a look at what needs to be done to create an AI runtime application. An AI application, like almost any basic application, consists of three major steps: input, processing and output. Different sources can be used as input, such as audio data, videos or time series values. Sensor-based time series values are ideal for performing tasks such as predictive maintenance, while audio data can be used, for example, to add a voice user interface to the machine. The input data must then often be pre-processed. Images often have to be cut together, sensor or audio data filtered.

The pre-processed data is then sent to the AI model, where the so-called inference is executed. In this context inference means to evaluate the already trained model to predict results from the input data. This is the real AI task; the rest of the application only serves to pre-process and post-process the data for this task. In the case of video data the inferencing happens on AI extension module. Inference from sensor data (other than image or video) usually does not require high computing power, therefore it can be computed directly on the CPU of the controller. The AI module is usually not required for such less computing-intensive applications. The results still need to be post-processed after the model inference finishes such that they can be used or saved in the main application and presented to a user.

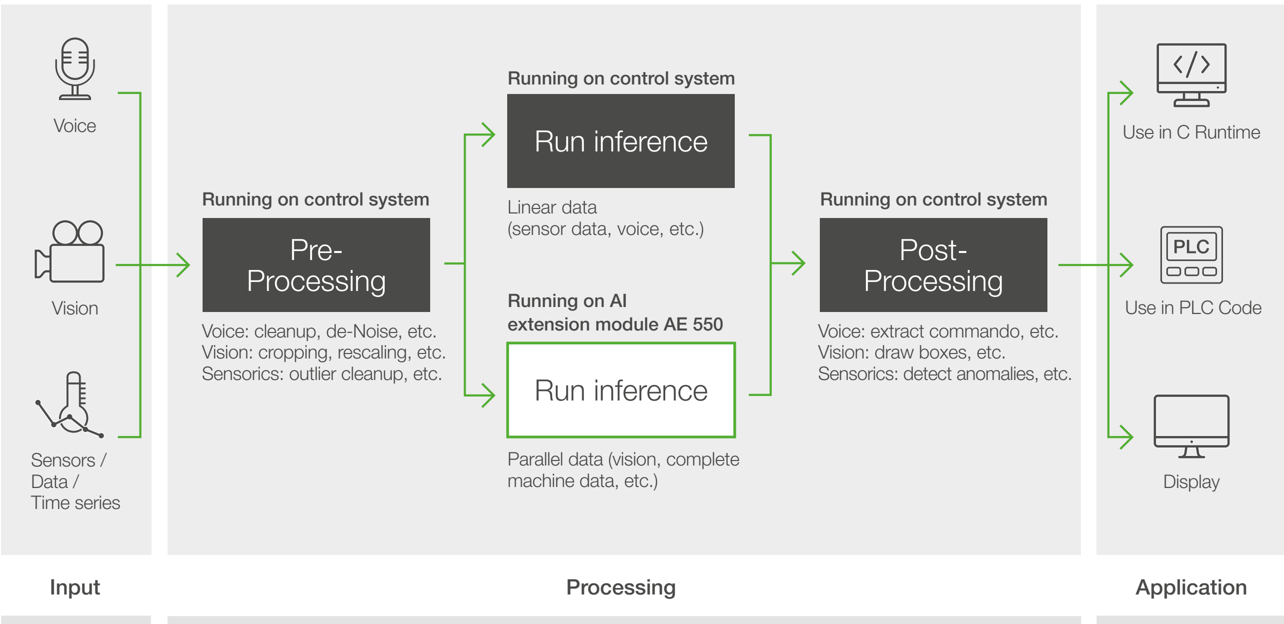

Figure 3 - Processing steps done in an AI runtime application. Sensor and/or vision data undergoes pre-processing. In the case of non-data-intensive tasks, the inference is done directly on the controller CPU, whereas vision data can benefit from the accelerated inference on the AI extension module (green box). The inference results are post-processed and sent to the main application.

Create an object tracking application

When creating an AI application, the first step is to identify and analyze the problem we want to solve with the application. In our robotics use case we want to detect chocolates and track their position over time on a conveyor belt. In the same process it is of course also possible to carry out an optical quality inspection of the chocolates. If this is desired, however, care must be taken when training the model to ensure that good and bad chocolates are made available to the training algorithm. If we know our problem, we can choose a model category and a model that we want to use to solve it.

Select & Train the AI Model

Since we want to detect chocolates in an image we need an object detection model. Such an AI model makes it possible to recognize different objects on which it has been trained before.

Choice of Camera and Compute Hardware

Next, we need to choose the appropriate camera for the application – this can be a 2D or 3D camera - and then adapt the lighting conditions to enhance image quality and accuracy. AI vision models are a lot more robust against changing light conditions than conventional vision algorithms, but of course they also need reasonably constant lighting to deliver reliable results. The Kemro X vision system supports many cameras that are connected either via USB or Ethernet, regardless of the camera manufacturer / brand.

The camera can therefore be selected by the user. If know-how in this area is lacking, KEBA offers support in this process as well as for pre-selected and pre-configured cameras for various use cases. For our use case mentioned above we decided to use an Industrial RGB Ethernet Camera with a resolution of 720x540 pixels with up to 290 frames per second. As we do not need to carry out optical quality control in this use case, we can use a camera with a lower resolution. The lower resolution allows us to transfer more images per second via the Ethernet interface. This means that even high-speed applications would be possible with this camera setup due to the many images per second the camera delivers. Of course the AI module is also capable of running an object detection model at this speed. If not so many pictures per second are required, this can simply be set in the camera settings. With this camera and a suitable lens, we have chosen one with a fixed focal length of 16mm, we are able to monitor the relevant part of the conveyor belt.

After that we need to decide which C5 control is to be used. A suitable control must be selected depending on the required computing power. However, if it is noticed at a later stage in the process that the required computing power is not sufficient, it is also possible to simply switch to a more powerful version. In general, it can be said that the more image processing operations have to be carried out, the more computing power is required for the control. For a simple application, as in our example, a control with medium computing power is completely sufficient. For our demo we use a KeControl CP507/C, which has a 4-core Intel Atom processor with 4GB RAM. This control has enough computing power to cover the AI as well as the PLC and robotics functions.

Create AI Vision Pipeline

Now, after we have selected our hardware and decided on an AI model, we can create the vision pipeline. This pipeline describes the “process” that is carried out with each individual image from the camera stream. The configuration of the pipeline is carried out directly in KeStudio.

There are function blocks for all required operations that are easy to parameterize and constitute the pipeline after linked to a sequence. The vision pipeline from our example is described in in more detail here.

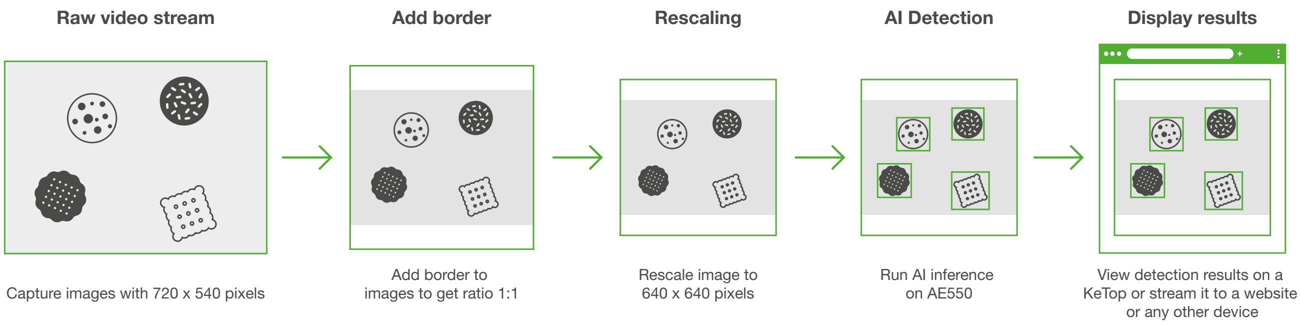

Figure 4 – Vision processing and AI inference steps done by the pipeline

The start of each pipeline is the reading of the raw image data from the camera. Camera settings such as image size, exposure, focus etc. can be made in a tool provided by the camera manufacturer or directly in KeStudio. In our example we grab images in RGB format with 720x540 pixels directly from the camera. The camera can also be triggered via digital input. We have wired this to a digital output from our control. This allows the camera to be triggered directly from the PLC program and receive the images at time triggered by the real-time application.

The next step is the pre-processing of the image. Possible operations here are cropping or rescaling the image or converting it to another pixel format if the camera cannot deliver the desired format. Computationally intensive operations are automatically outsourced by to the integrated GPU in the control unit to save CPU processing power. In our example we need to add a border to the image to get it in ratio 1:1 and then rescale it to the size of 640 by 640 pixels (which is required as input size by our AI model).

Object detection is then carried out on the pre-processed camera image. To do this, the image, and of course also the linked AI model, is sent to the AI extension module. The AI models can simply be loaded onto the control with KeStudio. They are then assigned to the corresponding function block in the program.

The recognition results can then be displayed on a panel or a website if desired. Videos can also be saved directly on the control. Of course, this is only possible if there is enough memory available. Normally, this is not done because videos consume several 100 MB of storage space quickly. This also depends on the resolution and the frames per second of the videos. This function was added for analysis or documentation purposes to save short excerpts.

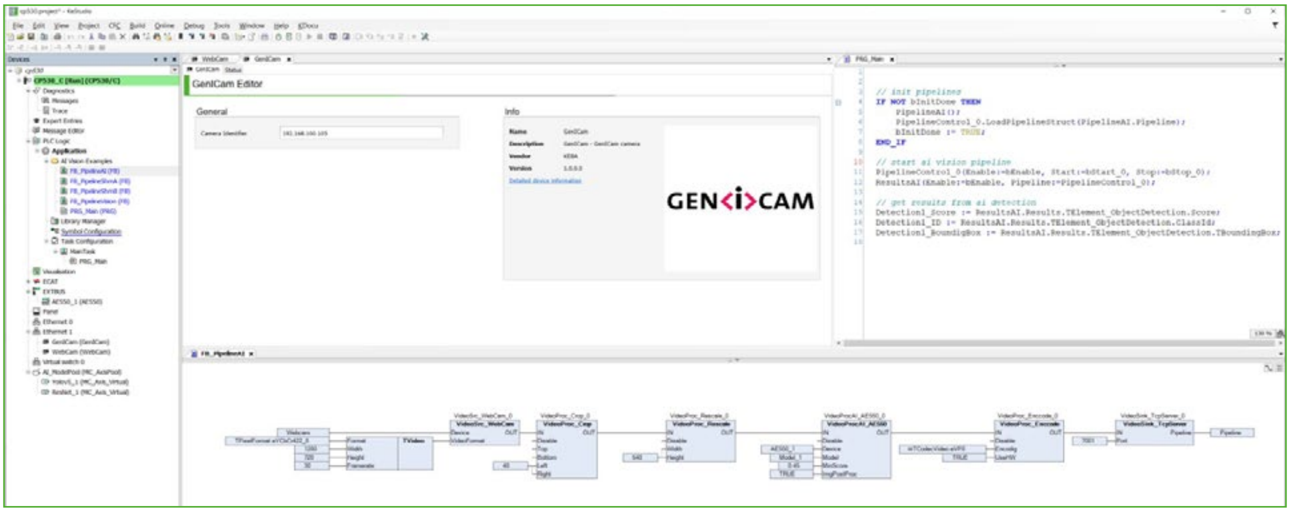

Figure 5 – Creation of a vision and AI pipeline directly in KeStudio. Cameras are configured in the device tree of the application (left and left top window) . Vision pipelines can be easily created with a function-block like editor (bottom window) and the results can be directly used in the IEC code (top left window)

Use AI Results in Main Application

Once the vision pipeline is created, it can be used in the main application. There is a separate function block that can be used to manage the pipeline. This can be done simply from text-based code in KeStudio. The results of the AI recognition can also be read out with a function block. This allows the user to access the positions of the object recognition directly in the PLC code and use them in the application, for example to have the robot grip the objects. The Kemro X vision system and AI extension module also support several simultaneous pipelines, for various tasks.

Conclusions

Creating a runtime application for object detection using the KEBA AE 550 AI extension module is streamlined due to its deep integration with the Kemro X control system. The process involves training the model, selecting appropriate camera and control hardware, configuring the vision pipeline which is used for interacting with the camera and the AI module. Finally, the vision pipeline is integrated into the main application, allowing easy access to AI detection results through PLC code function blocks. This integration simplifies the development and enhances the functionality of industrial applications.

Any questions?

Simply make an appointment. Our experts are guaranteed to work with you to find the perfect individual solution for your company.

Your Contact Person:
Stefan Fischereder
Productmanager Industrial AI
fsd@keba.com +43 732 7090 22723

News & Blog

Find the latest news, press releases and interesting facts in our blog:
www.keba.com/ia-blog-ai

KEBA Industrial Automation GmbH
Reindlstraße 51, 4040 Linz/Austria, Phone +43 732 7090-0, keba@keba.com

KEBA Group worldwide
Austria / China / Czech Republic / Germany / India / Italy / Japan / Netherlands / Romania /
South Korea / Switzerland / Serbia / Taiwan / Turkey / United Kingdom / USA

www.keba.com

Automation by innovation.