

Visionary AP Products Getting started

Mobile Perception

June 2025



SICK Visionary AP products

Visionary **AP products** are programmable **3D cameras** within the SICK AppSpace Ecosystem.

- **Made for:**

- 3D Vision specialists to develop and run customized Apps on the sensor
 - Utilize available AppSpace functions to solve highly sophisticated 3D vision applications
 - Deploy the solution on your or your customers devices

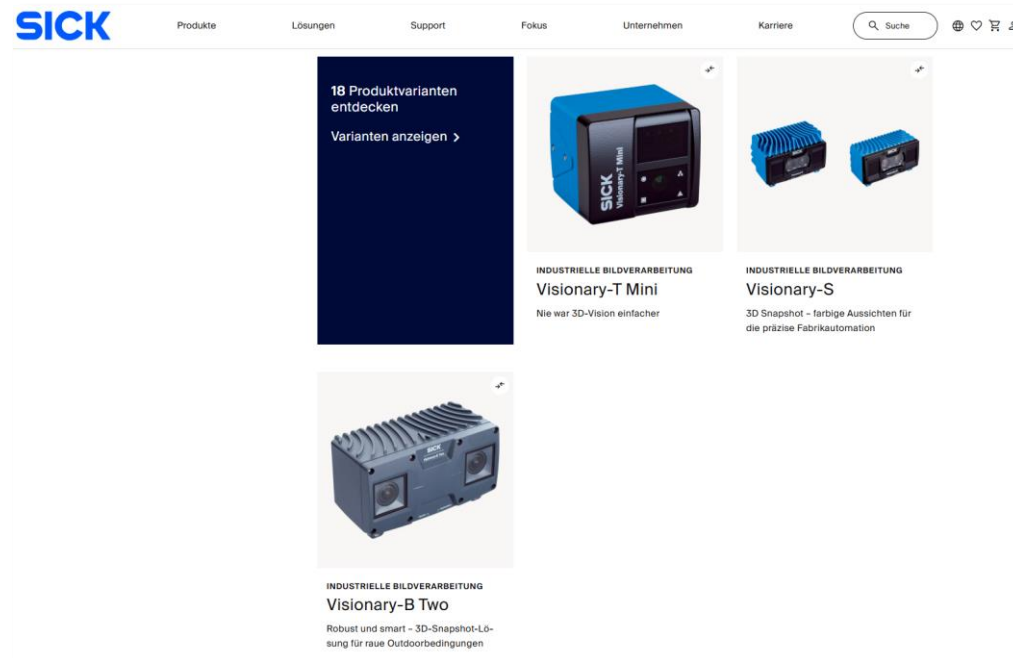


- Customers who want to purchase ready-made applications that solve a specific task and run on the sensor
 - configure the app for your specific application



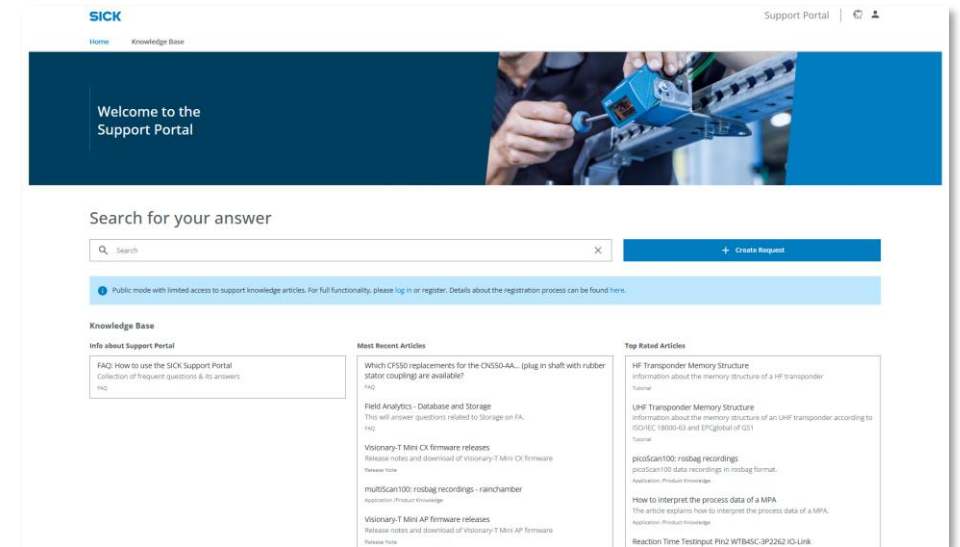
SICK Visionary AP products

- Choose from the programmable devices in the SICK Visionary portfolio
- Get an overview on all available Visionary AP devices on [sick.com](https://www.sick.com)



Where to find what

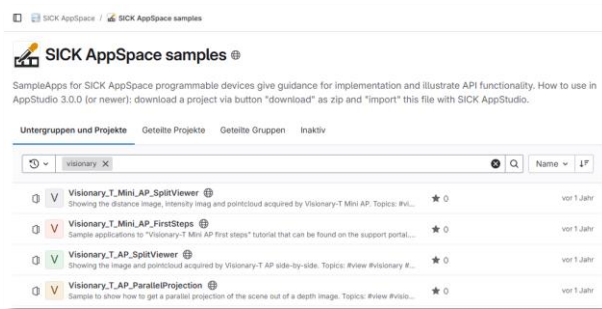
- Data sheets and documentation for the individual devices can be found on the corresponding device pages on [sick.com](https://www.sick.com)
- More information and support for the Visionary AP programmable devices is available on the [SICK Support Portal](#)
- Here you can find ...
 - [Support for general AppSpace topics and LUA coding](#)
 - Device specific release notes information (incl. API documentation, programming samples...)
 - [Visionary-S AP](#)
 - [Visionary-T Mini AP](#)
 - [Visionary-B Two](#)
 - [Help with your specific questions \(Ticket\)](#)



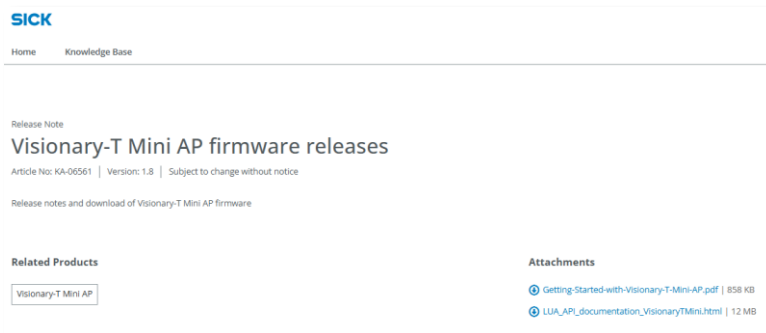
Where to find what

For each Visionary AP device, you can find ...

- Helpful [programming samples on GitLab](#) to get you started



- A full API documentation
 - Please find the API description among the attachments in the release notes



Software tools

The SICK AppManager is used to manage apps and firmware on Visionary AP devices

- Free of charge on www.sick.com
- [Documentation and support](#)



The SICK AppStudio is required to develop Apps

- Get a one-year-license on www.sick.com
- [Documentation and support](#)

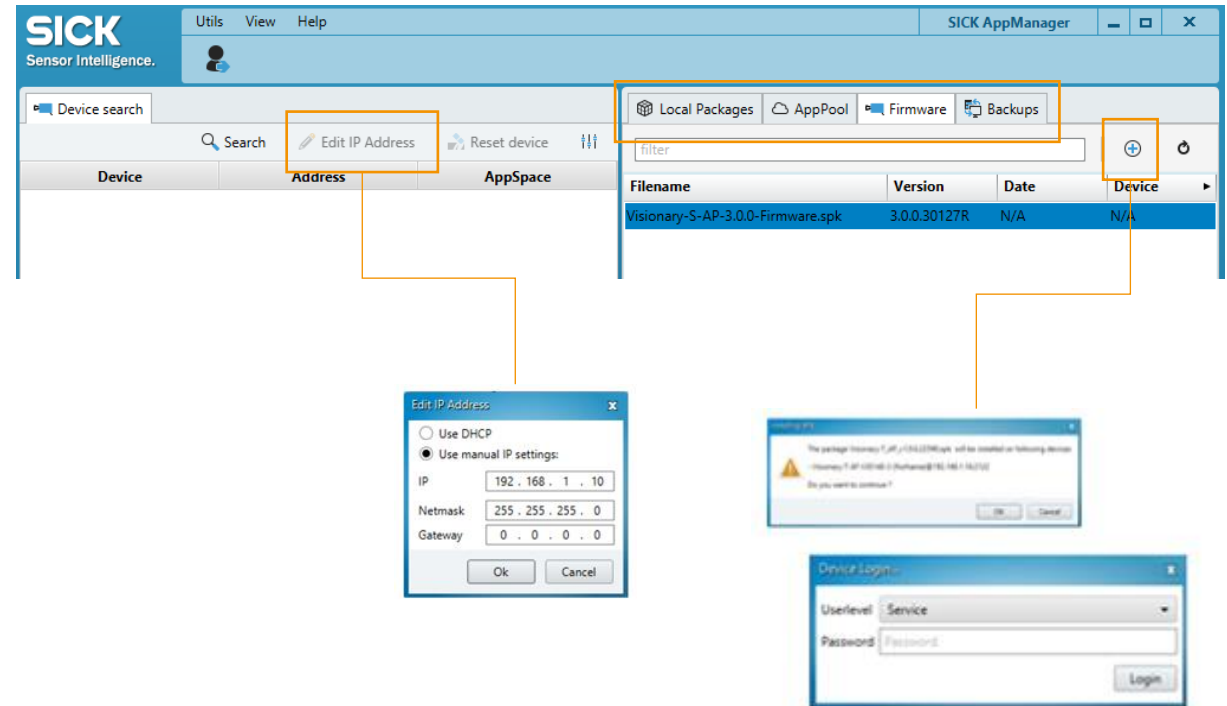


Manage Visionary AP devices

Use the **SICK AppManager** to manage Visionary AP devices

- Install apps to a device
 - own developed applications (deployed in AppStudio)
 - applications from the [SICK AppPool](#)
 - Ready-made SensorApps from [SICK](#)
- Update the device firmware
- Change the IP address of the device
- Create backups of a device

[Documentation and support](#)

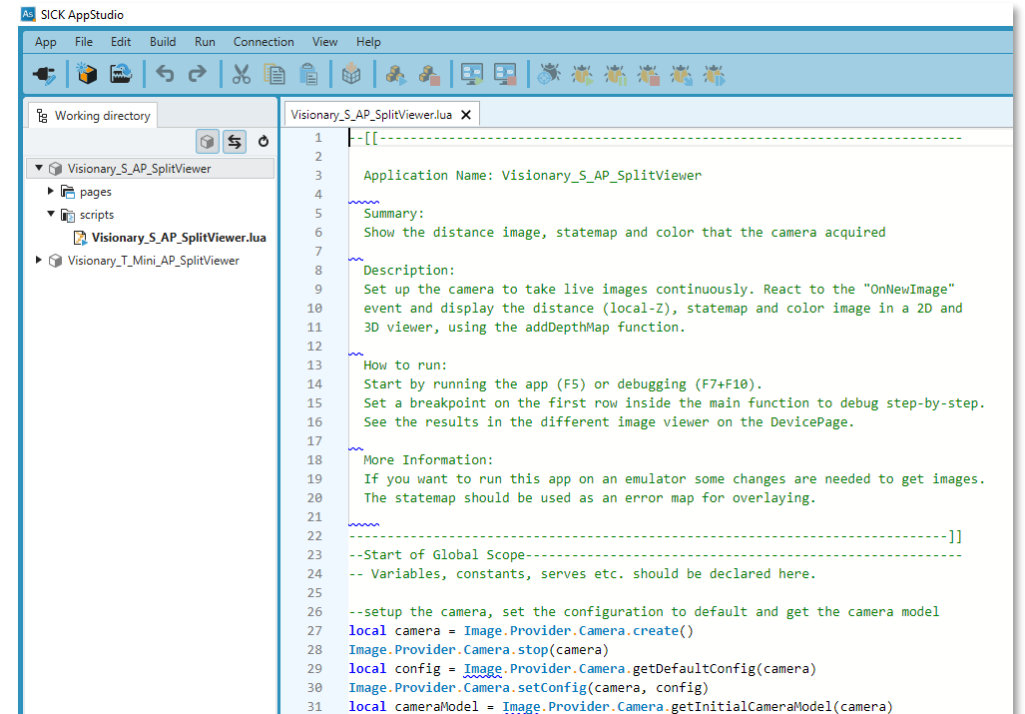


Program Visionary AP devices

Use the **SICK AppStudio** to develop your own applications

- SICK AppStudio is an Application development Kit within the AppSpace eco-system and provides
 - IDE (Integrated Development Environment) with editors and utilities
 - Emulators
- Use the available [programming examples](#) to get started
- Deploy applications that can be managed and redeployed by the AppManager

[Documentation and support](#)



The screenshot shows the SICK AppStudio IDE interface. The main window displays a Lua script for 'Visionary_S_AP_SplitViewer.lua'. The script includes a header with application name, summary, description, and instructions on how to run and debug the application. The code starts with a comment indicating the start of the global scope and includes variable declarations for camera, config, and cameraModel.

```
1  |-----|
2  |
3  | Application Name: Visionary_S_AP_SplitViewer
4  |
5  | Summary:
6  | Show the distance image, statemap and color that the camera acquired
7  |
8  | Description:
9  | Set up the camera to take live images continuously. React to the "OnNewImage"
10 | event and display the distance (local-Z), statemap and color image in a 2D and
11 | 3D viewer, using the addDepthMap function.
12 |
13 | How to run:
14 | Start by running the app (F5) or debugging (F7+F10).
15 | Set a breakpoint on the first row inside the main function to debug step-by-step.
16 | See the results in the different image viewer on the DevicePage.
17 |
18 | More Information:
19 | If you want to run this app on an emulator some changes are needed to get images.
20 | The statemap should be used as an error map for overlaying.
21 |
22 |-----|
23 | --Start of Global Scope-----|
24 | -- Variables, constants, serves etc. should be declared here.
25 |
26 |
27 | --setup the camera, set the configuration to default and get the camera model
28 | local camera = Image.Provider.Camera.create()
29 | Image.Provider.Camera.stop(camera)
30 | local config = Image.Provider.Camera.getDefaultConfig(camera)
31 | Image.Provider.Camera.setConfig(camera, config)
32 | local cameraModel = Image.Provider.Camera.getInitialCameraModel(camera)
```



Thank you for your attention.

SICK AG - Mobile perception