GstPluginPylon
A Study of Dynamic Element Properties in Basler Cameras

Miguel Taylor-Lopez

RidgeRun

Agenda

- Collaboration background
- Pylon and GenICam dynamic features
- The limitations of GObject Properties
- The alternative: Child Proxy
- Additional Solution Elements
- Outcome and impact
- Conclusions
- Questions

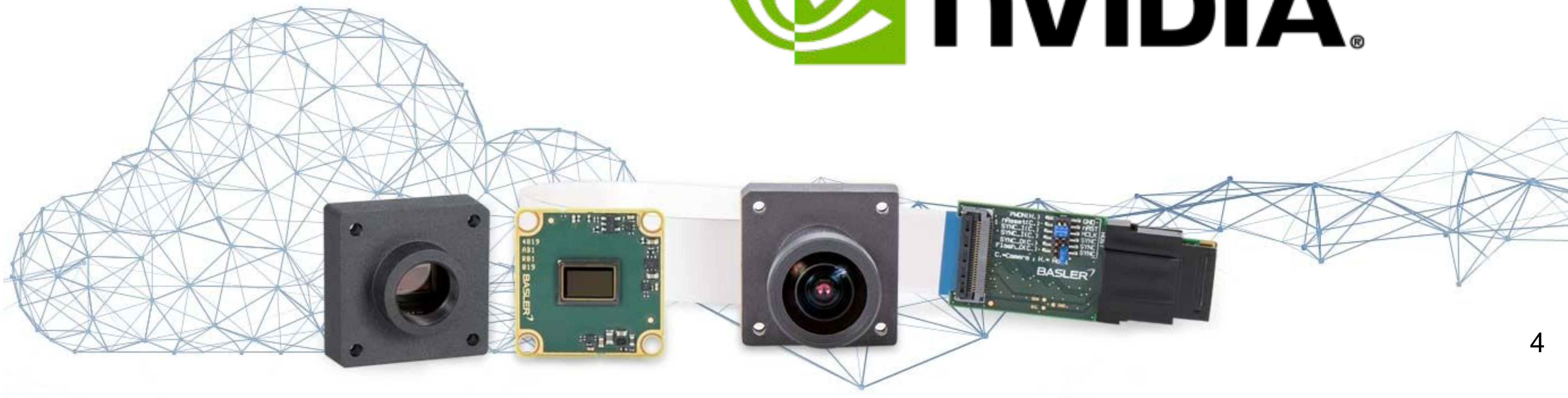
Collaboration Background

- RidgeRun is a software development and service integration company that specializes in embedded systems across various industries.
- Our areas of expertise include:
  - Embedded Linux.
  - Artificial Intelligence.
  - Computer Vision.
  - FPGA.
  - **GStreamer.**

- Basler is an internationally leading manufacturer of high-quality cameras and accessories for applications in:
  - Factory automation.
  - Medicine.
  - Traffic.
  - Other markets.

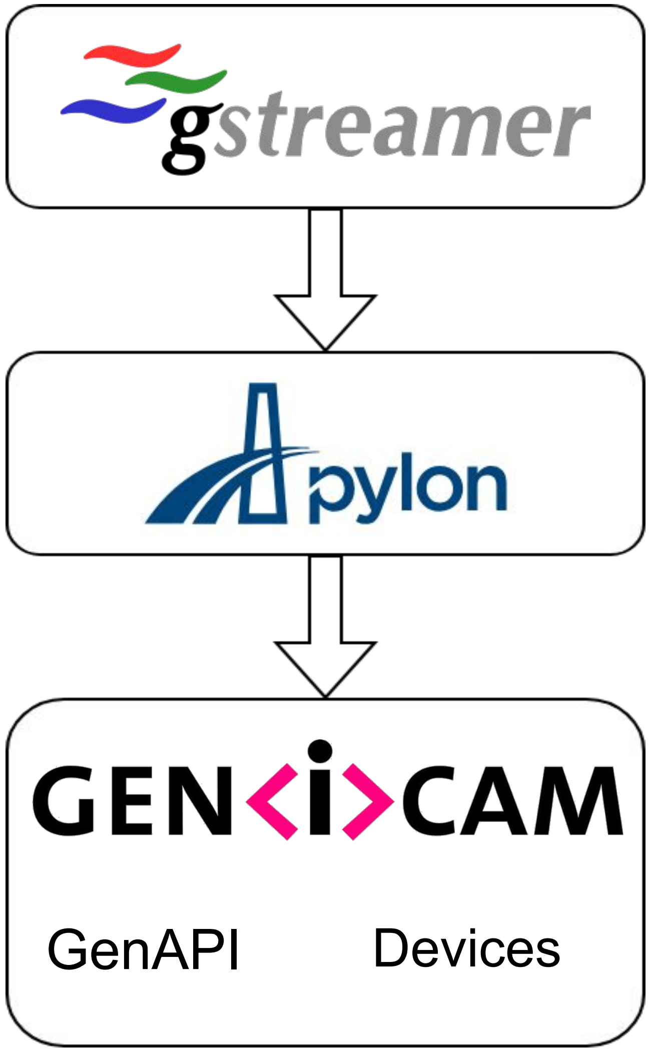
Collaboration Background

- RidgeRun partnered with Basler to develop the official GStreamer plug-in for Basler cameras
- Basler provides embedded cameras for NXP and NVIDIA devices, featuring USB3 and MIPI interfaces.



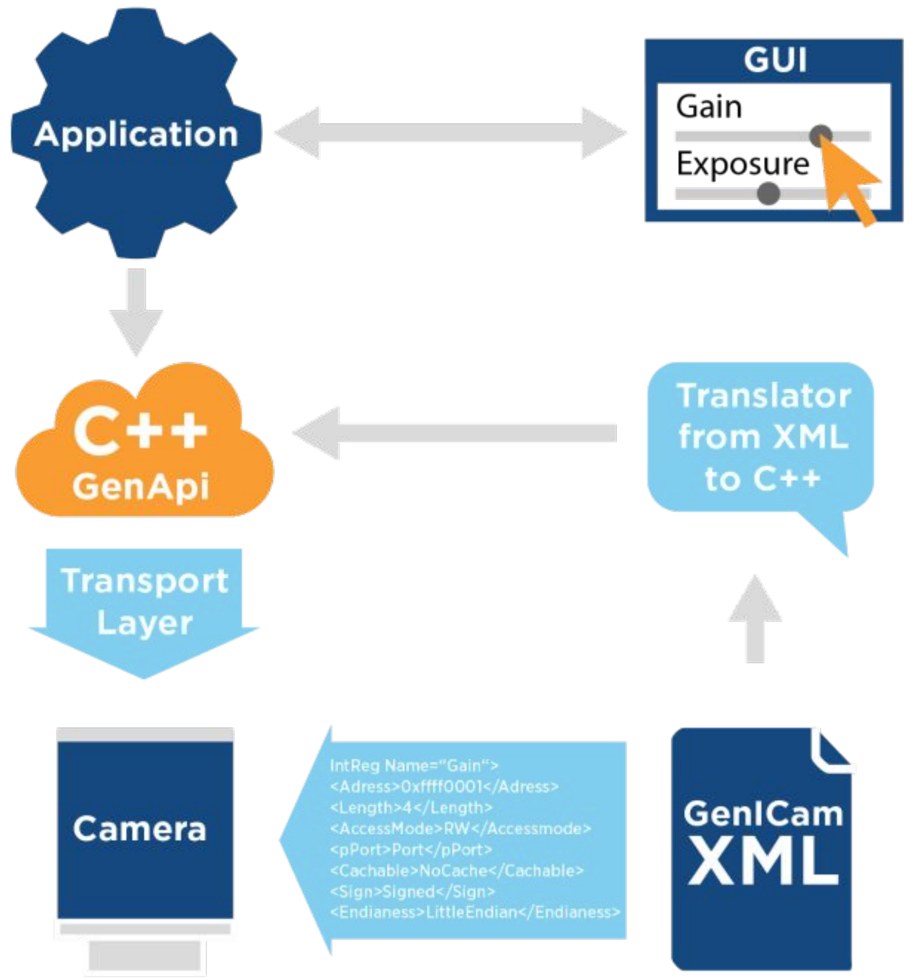
Pylon and GenICam dynamic features

- **Pylon SDK**: Basler uses Pylon SDK for camera inspection on runtime.
- **Dynamic camera Information**: With the SDK, we can obtain dynamic information from connected devices, including details like resolution, camera type, and more.
- **Integration Goal:** To incorporate dynamic camera discovery with pylon into Gstreamer.
- **Dynamic properties:** Depending on connected cameras, properties will be added to the source element to configure device-related features and interfaces.



The limitations of GObject Properties

- How is it done in v4l2src?
- **User-Friendly Approach:** Our goal was to create a user-friendly element by installing only supported features as properties, extracting property type, range, and default values from feature descriptions.
- **GObject Introspection:** Use GObject introspection for runtime object type determination.
- **Dynamic Device Discovery** when a new camera is connected.

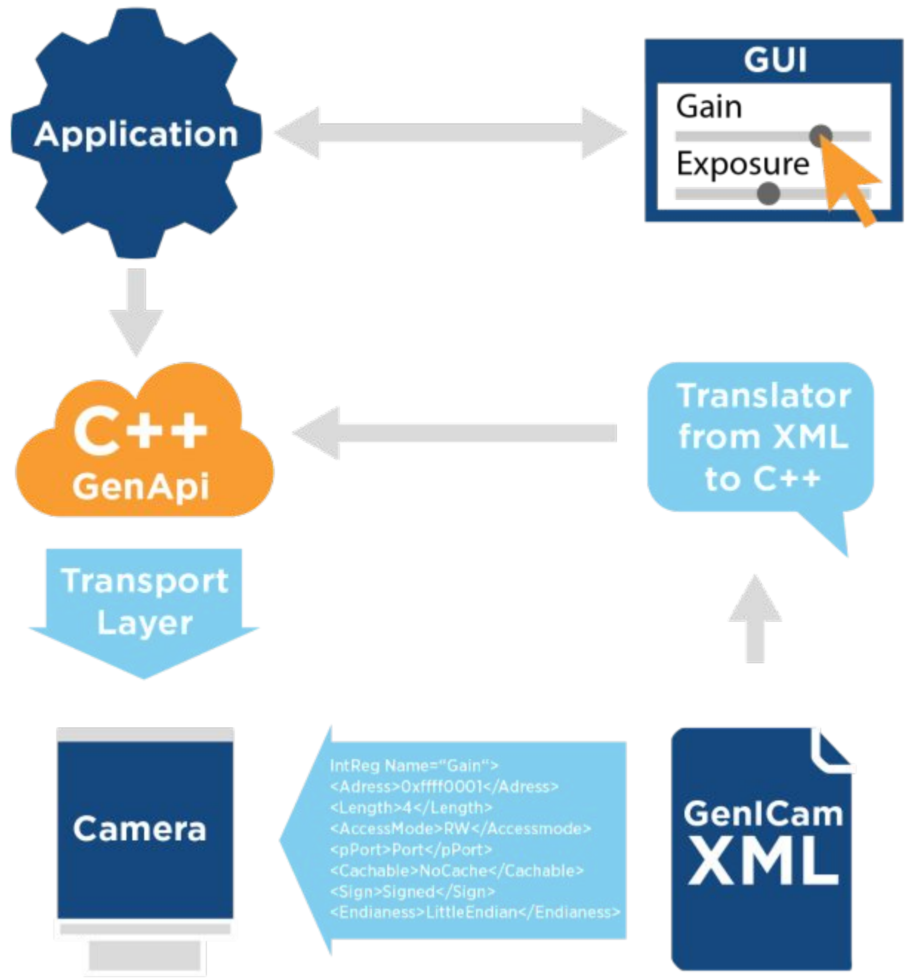


```
IntReg Name="Gain">
<Adress>0xffff0001</Adress>
<Length>4</Length>
<AccessMode>RW</Accessmode>
<pPort>Port</pPort>
<Cachable>NoCache</Cachable>
<Sign>Signed</Sign>
<Endianess>LittleEndian</Endianess>
```

The limitations of GObject Properties

Why can't we just simply inspect all the properties as regular GObject properties?

1. **Diverse Camera Features**
   - Cameras expose various features.
   - Not all properties apply universally.
2. **Name Ambiguity**
   - Multiple cameras share feature names.
   - Different definitions (ranges, types, etc.).
3. **Variable Access Permissions**
   - Cameras have different access flags.
   - Example: Gain may be writable only in certain cameras



```
IntReg Name="Gain">
<Adress>0xffff0001</Adress>
<Length>4</Length>
<AccessMode>RW</Accessmode>
<pPort>Port</pPort>
<Cachable>NoCache</Cachable>
<Sign>Signed</Sign>
<Endianess>LittleEndian</Endianess>
```

The alternative: Child Proxy

Using GstChildProxy: Our solution uses child proxy to handle different cameras.

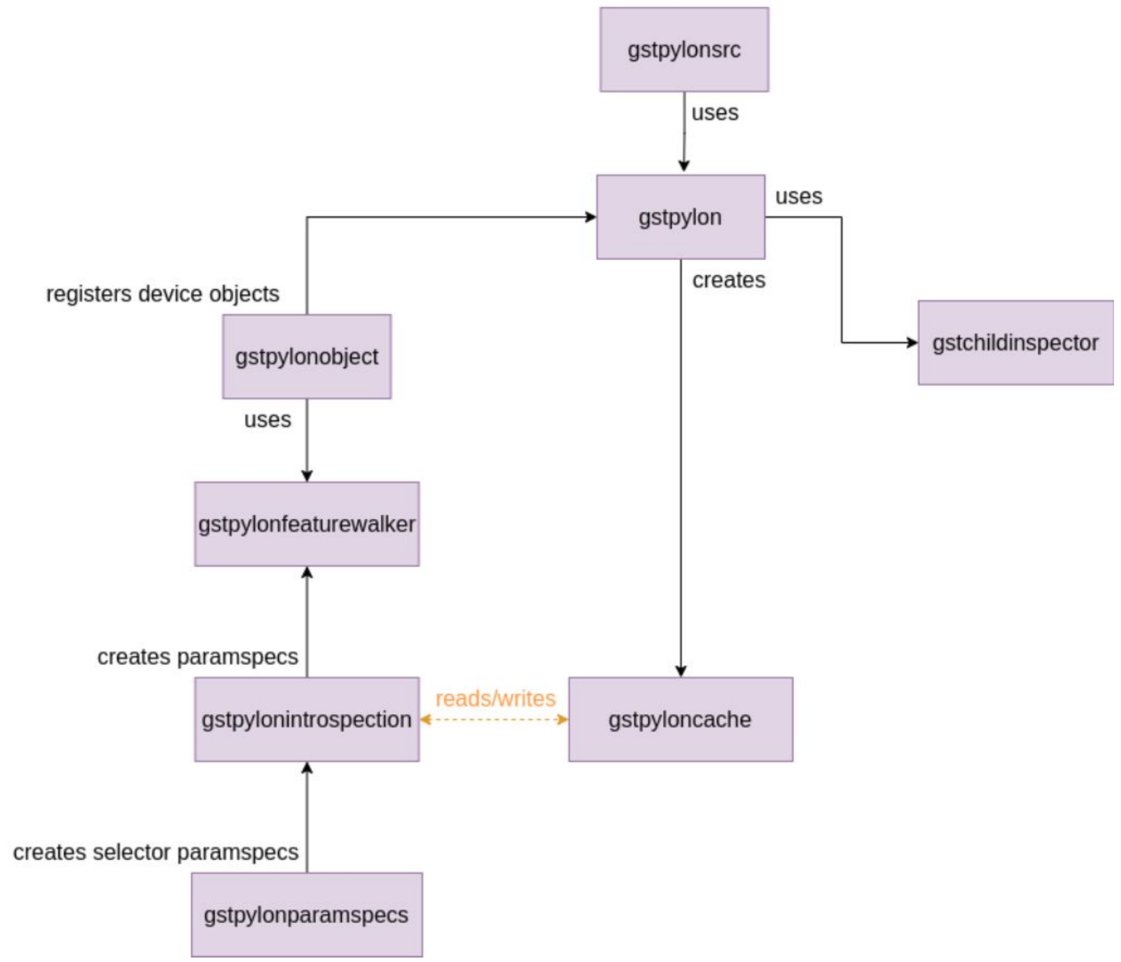
1. We installed a 'cam' property of type GObject.
2. The 'cam' property holds the currently active camera, which is a Pylon Device wrapped as a GObject.
3. Every Pylon device exposes its features as GObject properties.
4. We 'build' the description of the top 'cam' property dynamically by iterating through all the cameras, reading their properties, and appending their individual 'inspects'.
5. When the user selects a camera, it becomes available as the 'cam' child.
6. We use child proxy notation to modify a property in the actual camera.
7. This allowed us to have a 'stream' child with the same behavior.

Solution Design

Device registration flow

1. gstpylonsrc initialization
2. Device registration
3. Feature installation
4. GParamSpec generation
5. Custom GObject types

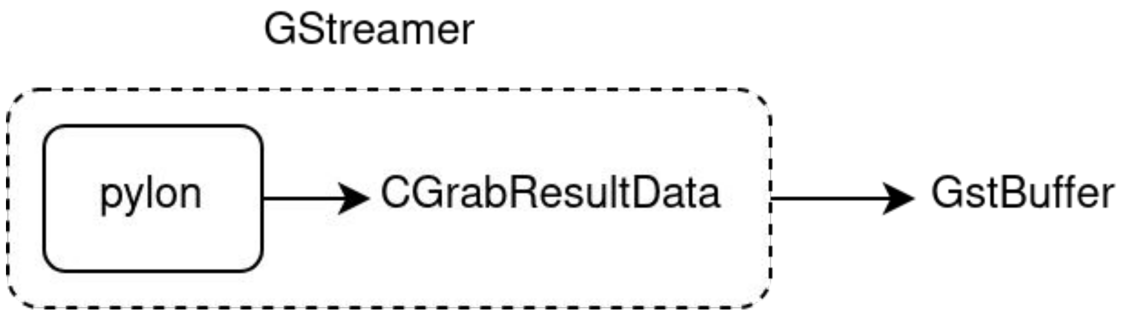
Element inspection enhancements: gstpyloncache



Additional Solution Elements

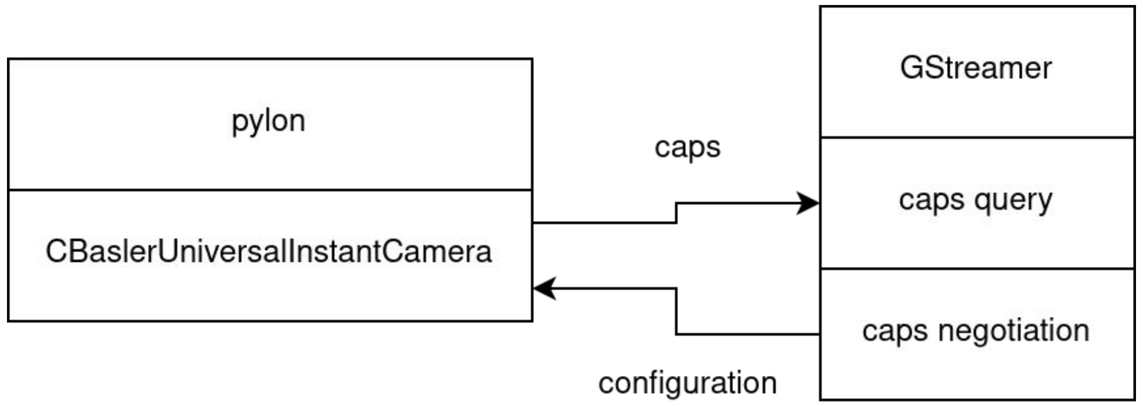
Memory Optimization

- The element creates a GstBuffer wrapper over Pylon data.
- Supports external buffer pools and implements a factory for NVMM and system mem buffers.



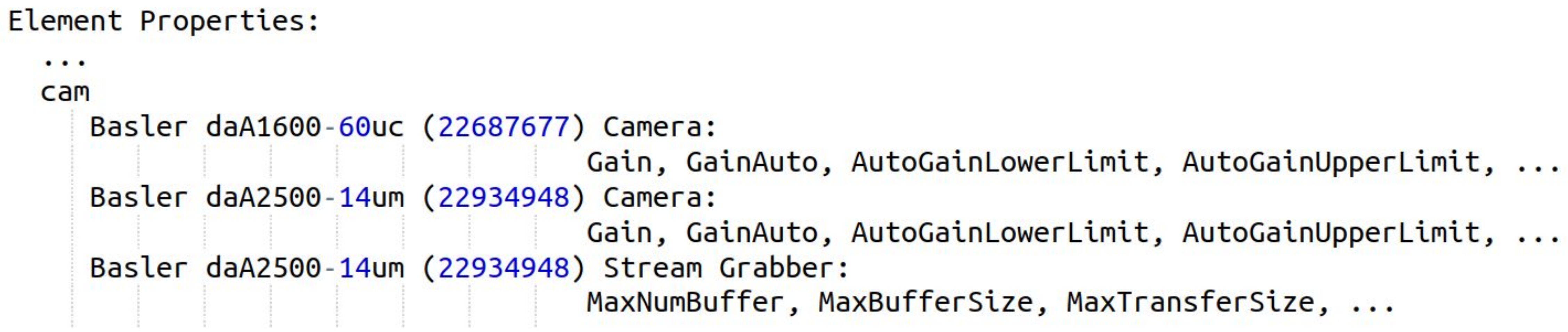
Caps negotiation

- An API call determines possible caps for a given device.
- GStreamer configures the resulting caps in the device using another API call.



Outcome and Impact

- **User-Friendly Element**: The result is an element that is user-friendly, allowing all camera features to be configurable at the user level and well-documented in "gst-inspect."
- **Developer-Friendly:** Well-organized and properly documented properties make it easier for developers to understand and maintain the codebase.
- **Dynamic Adjustments:** Users can easily change camera settings and configurations through property values, allowing for dynamic adjustments without modifying the source code or the pipeline.



```
Element Properties:
  ...
  cam
     Basler daA1600-60uc (22687677) Camera:
                                   Gain, GainAuto, AutoGainLowerLimit, AutoGainUpperLimit, ...
     Basler daA2500-14um (22934948) Camera:
                                   Gain, GainAuto, AutoGainLowerLimit, AutoGainUpperLimit, ...
     Basler daA2500-14um (22934948) Stream Grabber:
                                   MaxNumBuffer, MaxBufferSize, MaxTransferSize, ...
```

Outcome and Impact

Example: Selecting a camera

```
~$ gst-launch-1.0 pylonsrc device-name="Top-Left" device-index=0 !
autovideosink
```

Example: Changing a property

```
~$ gst-launch-1.0 pylonsrc device-index=0
cam::TestPattern=ColorDiagonalSawtooth8 ! autovideosink
```

Example: Load a configuration from a file

```
~$ gst-launch-1.0 device-index=0 pfs-file=defect_detection.pfs !
autovideosink
```

Conclusions and reference links

- We share a viable way to develop GStreamer element with dynamic properties using child proxy
- We achieved user-friendly, developer-friendly solution that enables dynamic camera control.
- Basler cameras and features implemented by RidgeRun can help NVIDIA Jetson and NXP i.MX8 users to bring up their dynamic video applications.

Links

- GitHub repo: https://github.com/basler/gst-plugin-pylon
- Post: https://www.ridgerun.com/post/official-gstreamer-plug-in-for-basler-cameras

GstPluginPylon A Study of Dynamic Element Properties in Basler Cameras

Miguel Taylor-Lopez

RidgeRun