Building DeepStream pipelines for Multiple Object Detection and Tracking

Photogrammetric and Robot Vision

David Conde Morales

University of Vigo

imcv
international master in
computer vision

Detection and Tracking

- **Detection** is the process of locating and identifying objects of interest in an image or a video frame.
- **Tracking** is the process of keeping the identification and following the movement and behavior of these objects over time across multiple frames.
- Detection and tracking can be performed **independently** or **jointly**. While most current detection approaches rely on Deep Learning, tracking may be performed afterwards.

Multiple Object Tracking

- **Multiple Object Tracking (MOT)** is a challenging task that requires detecting and identifying multiple objects of interest in a video.
- Involves different types of objects (e.g., pedestrians, cars) and different scenarios (e.g., surveillance, autonomous driving) that may have different characteristics and requirements.
- Requires efficient and robust algorithms that can handle large numbers of objects and high frame rates in real-time applications. Important for computation on the edge.

GStreamer — Introduction

GStreamer is a versatile framework written in C with GLib and GObject for creating streaming media applications:

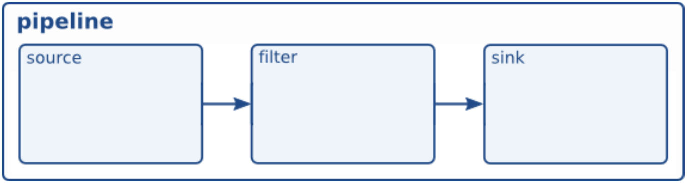
- Definition of interconnected elements (sources, filters and sinks).
- Cross-platform between various operating systems.
- Extensible with plugins for additional features, codecs and formats.

GStreamer — Elements

Pipelines simplest structures are called **Elements**, which may be of one of three types:

- **Source Elements:** Provide data.
- **Filter Elements:** Access/manipulate data.
- **Sink Elements:** Consume data.

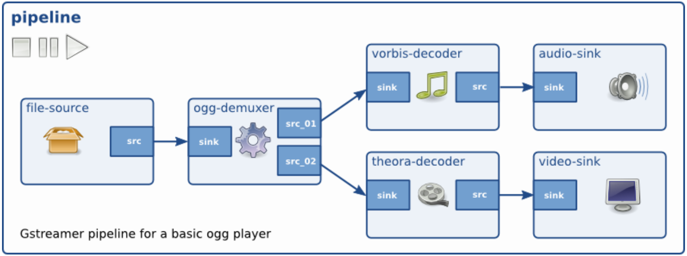
Some Elements may actually be **Bins**, which are containers of smaller Elements, while the general Bin is seen as an Element on its own. The Pipeline itself is considered as a special type of Bin.



GStreamer — Pads

The interface for each Element to communicate with others is defined by **Pads** with different availability type:

- **Always:** Always exist.
- **Sometimes:** Appear and disappear dynamically (e.g. demuxers).
- **On request:** Only when explicitly requested (e.g. tee elements).



Gstreamer pipeline for a basic ogg player

GStreamer — Capabilities

- The information that can travel through each Pad is defined by **Capabilities**.
- Each Pad can support multiple Capabilities to define specific behaviours (e.g. resolution, color space, framerate...), but when connected, Pads agree on a single common Capability via **negotiation**.
- Pads allow early refusal of connections between Elements if Capabilities don't intersect.

Capability example

- `video/x-raw,width=1280,height=720`

GStreamer — CLI Tools

Simple pipelines can be constructed with some pre-built CLI tools installed with the framework:

- `gst-inspect-1.0` takes as attribute a single Element name to display its information.
- `gst-launch-1.0` can be used to build on-the-fly a new Pipeline introducing Elements defined in `GST_PLUGIN_PATH`. Elements are connected with the ”**!**” operator, including Capabilities definitions in-between when required.

gst-launch-1.0 example

```
gst-launch-1.0 videotestsrc !
video/x-raw,width=1280,height=720 !  autovideosink
```

DeepStream — Introduction

DeepStream is an SDK designed for creating vision AI applications:

- Built upon the GStreamer framework.
- Includes its own set of GStreamer plugins for AI inference, video analytics, and real-time processing.
- Allows adding custom metadata and easily optimizing supported models with **TensorRT**.

DeepStream — Installation and Demo

- Downloadable resources maintained in official NGC Catalog:
  - dGPU.
  - Jetson.
- Windows Subsystem for Linux (WSL).

Hands-on course

Follow the instructions in this repository