USE DEEPSTREAM AND THE TLT TO DEPLOY STREAMING ANALYTICS AT SCALE
AGENDA

1. Real-time Streaming for Video Analytics
2. Framework for Analyzing Video
3. Understand the Basics: DeepStream SDK 3.0
4. Build with DeepStream: Example Applications
5. NVIDIA Transfer Learning Toolkit
REALTIME STREAMING VIDEO ANALYTICS
REALTIME STREAMING VIDEO ANALYTICS

- Access Control
- Managing operations
- Parking Management
- Traffic Engineering
- Retail Analytics
- Optical Inspection
- Managing Logistics
- Content Filtering
DEPLOY FROM EDGE TO CLOUD

EDGE AND ON-PREMISES

- Camera
- NVR
- Server

CLOUD

- Data center

TENSORRT • DEEPSTREAM

- JETSON
- TESLA
FRAMEWORK FOR ANALYZING VIDEO
REALTIME STREAMING VIDEO ANALYTICS
UNDERSTAND THE BASICS:
DEEPSTREAM SDK 3.0
### DEEPSTREAM SDK 3.0

#### Plugins (build with open source, 3rd party, NV)
- DNN inference/TensorRT plugins
- Communications plugins
- Video/image capture and processing plugins
- 3rd party library plugins ... ...

#### Analytics - multi-camera, multi-sensor framework
- DeepStream in containers, Multi-GPU orchestration
- Tracking & analytics across large scale/ multi-camera
- Streaming and Batch Analytics
- Event fabric

#### Development Tools
- End to end reference applications
- App building/configuration tools
- End-end orchestration recipes & adaptation guides
- Plugin templates, custom IP integration

---

#### DeepStream SDK

- TensorRT
- Multimedia APIs/Video Codec SDK
- Imaging & Dewarping library
- Metadata & messaging
- NV containers
- Message bus clients
- Multi-camera tracking lib

#### Linux, CUDA

- Perception infra - Jetson, Tesla server (Edge and cloud)
- Analytics infra - Edge server, NGC, AWS, Azure
WHAT’S NEW IN DEEPSTREAM 3.0

LATEST PLATFORMS - TESLA
T4, Jetson XAVIER

TensorRT 5, CUDA 10

NEW PLUGINS

PLUGIN
LOW LEVEL LIB
GPU

Increased capability and throughput

CONNECT EDGE TO CLOUD

Stream and Batch Analytics on Metadata

EASY TO SCALE AND MANAGE

Deploy in Docker Containers

DYNAMIC STREAM MANAGEMENT

Add, remove, modify streams on the fly

LATEST PLATFORMS - TESLA
T4, Jetson XAVIER

TensorRT 5, CUDA 10

NEW PLUGINS

PLUGIN
LOW LEVEL LIB
GPU

Increased capability and throughput

CONNECT EDGE TO CLOUD

Stream and Batch Analytics on Metadata

EASY TO SCALE AND MANAGE

Deploy in Docker Containers

DYNAMIC STREAM MANAGEMENT

Add, remove, modify streams on the fly

HIGH EFFICIENCY AND THROUGHPUT WITH
Transfer Learning Toolkit

Transfer Learning Toolkit model files are plug-n-play

Dynamic stream management and containers available on Tesla only. Jetson support coming soon
GSTREAMER

<table>
<thead>
<tr>
<th>Level</th>
<th>Component</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PLUGINS</td>
<td>Basic building block connected through PADs</td>
</tr>
<tr>
<td>2</td>
<td>BINS</td>
<td>A container for a collection of plugins</td>
</tr>
<tr>
<td>3</td>
<td>PIPELINE</td>
<td>Top level bin providing a bus and managing the synchronization</td>
</tr>
</tbody>
</table>
**GSTREAMER**

<table>
<thead>
<tr>
<th>Object</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffers</td>
<td>Pass streaming data between plugins in the pipeline</td>
</tr>
<tr>
<td>Events</td>
<td>Send info between plugins or from the application to plugins</td>
</tr>
<tr>
<td>Messages</td>
<td>Post info on message bus for collection by the application</td>
</tr>
<tr>
<td>Queries</td>
<td>Allow applications to request information from the pipeline</td>
</tr>
</tbody>
</table>

**Gstreamer**

- **Application**
  - events
  - queries

- **Bus**
  - messages

- **Pipeline**
  - Video File
  - Decoder
  - Scale
  - Filter
  - Display
  - buffers
  - buffers
  - buffers
  - buffers
DEEPSTREAM BUILDING BLOCK

- A plugin model based pipeline architecture
- Graph-based pipeline interface to allow high-level component interconnect
- Heterogenous processing on GPU and CPU
- Hides parallelization and synchronization under the hood
- Inherently multi-threaded
Efficient Memory Management

DeeStream App

nvdec_h264

nvinfer (PGIE)

nvtracker

nvinfer (SGIE)

CUDA HW Buffer sharing

cudaMalloc

Returns CUDA HW Buffer for reuse

Gstreamer Plugin

Allocates CUDA HW Buffers (GPU Memory)

Uses shared CUDA HW (GPU Memory) buffers
MEMORY MANAGEMENT

GPU to CPU Copy

DeepStream App

gst-nvdecode

Gst-nvinfer

Gst-nvvidconv

Gst-xvimagesink

cudaMalloc

Efficient Buffer Copy
METADATA STRUCTURE

- **NvDSObjectParams** - Contains a subset of metadata information for an object detected in the frame.
- **GIE_Unique_ID** - Multiple neural networks get assigned a unique ID.
- **Num_rects** - Number of objects detected in the frame.
- **Stream_Id** - In case of multi-stream, to identify we need a stream id to associate which stream the data belongs to.
METADATA STRUCTURE

- **NvOSD_RectParams** - Bounding box coordinates
- **NvOSD_TextParams** - Label information required for display (white car, Mercedes, sedan)
- **NvDSAttribinfo** - Attributes of objects (type, color, make)
- **Tracking_ID** - Unique ID of that object from tracker
- **Class_ID** - Type of object (person, vehicle, two-wheeler, road sign)
DYNAMIC STREAM MANAGEMENT

Application

1. Add / Remove camera streams
2. Change FPS
3. Change resolutions
SCALE WITH DEEPSTREAM IN DOCKER

CONTAINER 1

DeepStream Applications
DeepStream SDK
TensorRT
CUDA Toolkit
Video Codec SDK
Container OS User Space
Docker Engine
CUDA Driver
Host OS
NVIDIA GPUs
Server

CONTAINER N

nvidia. GPU CLOUD

Discover GPU-Accelerated Containers

Innovate in Minutes, Not Weeks

Stay Up to Date

DECODER PLUGIN

gst-nvvideocodecs

Input: H.264, H.265, VP8, VP9, MPEG2/4

Output: NV12

Parameters: Bit rate control, i-frame decoding
BATCHING - GSTNVSTREAMMUX

**UPSTREAM**
- Buffers Input
  - Decoder / Camera 1
  - Decoder / Camera 2
  - Decoder / Camera...

**BATCHING**
- Batch of N buffers of size (W x H)
- Batched
- Round robin
  - Scaling

**DOWNSTREAM**
- NV12/RGBA Batched Buffer and NvStreamMeta
**GSTNVINFER**

**PREPROCESSING**
- Convert and/or Scale to network input resolution & color format
- Mean subtraction and scaling
- Optionally crop primary detected objects for secondary inferencing

**TensorRT IPlugin implementation for custom layers**
**Custom BBOX parsing function**

Lib containing IPlugin implementation

Unmodified input buffer

NV12 / RGBA Buffers (Unbatched)
OR Buffers (Batched) + NvStreamMeta

Unmodified input metadata, NvDsMeta attached by this element

TRT Inference

Bounding box parsing for clustering

ENABLING 360D CAMERA PROCESSING

NVWARP360 SDK

Equirectangular

Cylindrical

Rotated cylinder

Panini

Pushbroom
METADATA TO MESSAGE BROKERS

gst-nvmsgtransform & gst-nvmsgbroker

<table>
<thead>
<tr>
<th>Input</th>
<th>Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Message sent over supported protocol</td>
</tr>
</tbody>
</table>

Parameters

- Path to shared library implementing message generation from metadata based on schema
- Protocol, URL, port, topic for message destination
- Path to shared library implementing adaptor for desired protocol
# NVIDIA-ACCELERATED PLUGINS

<table>
<thead>
<tr>
<th>Plugin Name</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>gst-nvvideocodecs</td>
<td>Accelerated video decoders</td>
</tr>
<tr>
<td>gst-nvstreammux</td>
<td>Stream aggregator - muxer and batching</td>
</tr>
<tr>
<td>gst-nvinfer</td>
<td>TensorRT based inference for detection &amp; classification</td>
</tr>
<tr>
<td>gst-nvtracker</td>
<td>Reference KLT tracker implementation</td>
</tr>
<tr>
<td>gst-nvosd</td>
<td>On-Screen Display API to draw boxes and text overlay</td>
</tr>
<tr>
<td>gst-tiler</td>
<td>Renders frames from multi-source into 2D grid array</td>
</tr>
<tr>
<td>gst-eglgllessink</td>
<td>Accelerated X11 / EGL based renderer plugin</td>
</tr>
<tr>
<td>gst-nvvidconv</td>
<td>Scaling, format conversion, rotation</td>
</tr>
<tr>
<td>Gst-nvdewarp</td>
<td>Dewarping for 360 Degree camera input</td>
</tr>
<tr>
<td>Gst-nvmsgconv</td>
<td>Meta data generation</td>
</tr>
<tr>
<td>Gst-nvmsgbroker</td>
<td>Messaging to Cloud</td>
</tr>
</tbody>
</table>
CONFIGURATION FILE

enable-perf-measurement=1  //To enable performance measurement
perf-measurement-interval-sec=10  //Sampling interval in seconds for performance metrics
flow-original-resolution=1  //Stream muxer flows original input frames in pipeline
#gie-kitti-output-dir=/home/ubuntu/kitti_data/  // location of KITTI metadata files

[source0]
enable=1  // Enables source0 input
#Type - 1=CameraV4L2 2=URI 3=MultiURI  //1) Input source can be USB Camera (V4L2)
 // 2) URI to the encoded stream. Can be a file, HTTP URI or an RTSP live source
 // 3) Select URL from multi-source input
type=3  // Type of input source is selected
uri=file://../..//streams/sample_720p.mp4  // Actual path of the encoded source.
num-sources=1  // Number of input sources.
gpu-id=0  // GPU ID on which the pipeline runs within a single system

MULTI-STREAM REFERENCE APPLICATION
SMART PARKING SOLUTIONS WITH DEEPTREAM
PERCEPTION AND ANALYTICS IN CONTAINERS
SCALING AND PORTABILITY

PERCEPTION SERVER

DeepStream

DeepStream

Metadata

Metadata

Message Broker

Kafka

Message Broker

Kafka

Python Module

Anomaly Detection

Apache Spark

Occupancy State

ETL

Logstash

Multi Camera

Tracking / Smoothing

Parking Garage

State Mgmt

Search Indexer

ElasticSearch

Kafka

Metadata

Analytics Server

Multi Camera

Tracking / Smoothing

Python Module

Parking Garage

State Mgmt

Search Indexer

ElasticSearch

Kibana

API

Browser Apps

Kafka

Metadata

DeepStream

DeepStream

Analytics Server

Python Module

Logstash

ETL

Anomaly Detection

Occupancy State

Parking Garage

State Mgmt

Search Indexer

ElasticSearch

Kibana

API

Browser Apps

Kafka

Metadata
COMMAND CENTER UI

Browser-Based Interface

- Occupancy Map for Multiple Levels
- Sensor Fusion
- Anomalies
- Search Events and Anomalies
- Occupancy Stats and Flow Rates
ANALYTICS DASHBOARD

Spot Occupied: 1,204
Available Spots: 53

Endeavor-Occupancy%: Spot Occupied 96% / 1,257

Endeavor-AnomalyCountByType:

Endeavor-EntryExitHeatmap:
NVIDIA TRANSFER LEARNING TOOLKIT
IVA DEEP LEARNING WORKFLOW MANAGEMENT

Third Party Pre-Trained Models
- Lack accuracy
- Use case limitations
- Model size limitations
- Unoptimized for GPUs

Deep Learning Training
- Compute Resources
- Time spent training from scratch
- Learning DL frameworks

Deep Learning Inference
- Unclear workflows for production ready models
- Complex application pipeline

Accelerate deep learning training with pre-trained models and functions provided by Transfer Learning Toolkit
Transfer Learning Workflow

Fine Tuning * Pruning * Scene Adaptation * New Classes

Output Model ready to be deployed and integrated for us with DeepStream SDK 3.0 applications
Workflow With DeepStream

- Train with new data
- Adapt to scene
- Add new classes
- Prune the models
- Export in a DeepStream application

Transfer Learning Toolkit for IVA

DeepStream Application
End to End Deep Learning Workflow

Pre-Trained models * Training & Adaptation pipeline * Applications ready for DeepStream integration

Accelerate time to market and save on compute resources!
FEATURES

Efficient Pre-trained Models
GPU-accelerated models trained on large scale public datasets.

Faster Inference with Model Pruning
Model pruning reduces size of the model resulting in faster inference.

Training with Multiple GPUs
Re-training models, adding custom data for multi GPU training using an easy to use tool.

Abstraction
Abstraction from having deep knowledge of frameworks, simple intuitive interface to the features.

Containerization
Packaged in a container easily accessible from NVIDIA GPU Cloud website. All code dependencies are managed automatically.

Integration
Models exported using TLT are easily consumable for inference with Deep Stream SDK.
START DEVELOPING WITH NVIDIA TRANSFER LEARNING TOOLKIT AND DEEPSTREAM

DEEPSTREAM . TRANSFER LEARNING TOOLKIT . SUPPORT FORUMS