An efficient workflow for portable computer vision algorithms using OpenVX

Niclas Danielsson, Mikael Pendse
Axis Communications AB
Axis at a glance

3,253 employees* with presence in more than 50 countries

Total sales 2018 1,2B USD**

Part of Canon Group

Local presence around the world through extensive partner network

1,2B USD**

15 years of steady growth

82% highly satisfied partners

Axis Group companies

* Q4 2018 **Full year 2018 SEK 10.3 B EUR 1,0 B
Solutions for a smarter, safer world

**Video surveillance solutions**
Cameras, encoders, VMS, recorders, analytics and applications.
Ensure scalability and simplify integration.

**Access control solutions**
From identification and entry control to advanced access management, open access control solutions integrate seamlessly with other systems.

**Audio solutions**
Complete, high-quality audio systems.
Filling the need for callouts in security solutions, background music and paging in any setting.

**Training, service & support**
Academy training courses, certification, technical support and tools.

www.axis.com
> Acceleration
  - Proprietary vision accelerator on our ARTPEC chips used in our video motion detection.

> Portability
  - Axis uses many different chip solutions. Standardization is a must for portability.
"Hello OpenVX" graph example

```c
vx_context context = vxCreateContext();
vx_image images[] = {
    vxCreateImage(context, 640, 480, VX_DF_IMAGE_UYVY),
    vxCreateImage(context, 640, 480, VX_DF_IMAGE_U8),
    vxCreateImage(context, 640, 480, VX_DF_IMAGE_U8),
};

vx_graph graph = vxCreateGraph(context);
vx_image virts[] = {
    vxCreateVirtualImage(graph, 0, 0, VX_DF_IMAGE_VIRT),
    vxCreateVirtualImage(graph, 0, 0, VX_DF_IMAGE_VIRT),
    vxCreateVirtualImage(graph, 0, 0, VX_DF_IMAGE_VIRT),
    vxCreateVirtualImage(graph, 0, 0, VX_DF_IMAGE_VIRT),
};

vxChannelExtractNode(graph, images[0], VX_CHANNEL_Y, virts[0]),
vxGaussian3x3Node(graph, virts[0], virts[1]),
vxSobel3x3Node(graph, virts[1], virts[2], virts[3]),
vxMagnitudeNode(graph, virts[2], virts[3], images[1]),
vxPoseNode(graph, virts[2], virts[3], images[2]),

status = vxVerifyGraph(graph);
if (status == VX_SUCCESS)
{
    status = vxProcessGraph(graph);
}
vxReleaseContext(&context); /* this will release everything */
```
Video motion detection graph example

```c
function_node = vxSubtractNode(graph_skeleton, input_images[1], input_images[2], VX_CONVERT_POLICY_SATURATE, internal_images[2]);
vxReleaseNode(function_node);
function_node = vxAddNode(graph_skeleton, internal_images[1], internal_images[2], VX_CONVERT_POLICY_SATURATE, internal_images[3]);
vxReleaseNode(function_node);

multiply_value = 1;
multiply_scalar = vxCreateScalar(graphmanager_get_context(graph_manager), VX_TYPE_FLOAT32, multiply_value);
function_node = vX eSportsNode(graph_skeleton, internal_images[1], internal_images[2], VX_CONVERT_POLICY_SATURATE, VX_ROUND_POLICY_TO_NEAREST_EVEN, internal_images[2]);
vxReleaseScalar(multiply_scalar);

depth_scalar = vxCreateScalar(graphmanager_get_context(graph_manager), VX_TYPE_INT32, 4; depth_value);
function_node = vxConvertDepthNode(graph_skeleton, internal_images[1], internal_images[2], VX_CONVERT_POLICY_SATURATE, depth_scalar);

vxReleaseScalar(depth_scalar);
function_node = vxReleaseNode(function_node);

depth_value = 1;

border_mode = vx-border_t border_mode, sizeof(vx_border_t)
node_rc_create[4](node_rc_create, graph_skeleton);
node_rc_release[4](node_rc_release);
dynamic_node[1] = node_rc_copy_ref[node_node_rc];

border_mode = VX_BORDER_REPLICATE;

vxSetNodeAttribute(node_node, VX_NODE_BORDER, &border_mode, sizeof(vx_border_t));
node_rc_release[4](node_rc_release);

depth_value = 1;

border_mode = VX_BORDER_REPLICATE;

vxSetNodeAttribute(node_node, VX_NODE_BORDER, &border_mode, sizeof(vx_border_t));
node_rc_release[4](node_rc_release);

vxReleaseNode(node_node);

vxReleaseNode(node_node);

vxReleaseNode(node_node);

vxReleaseNode(node_node);

vxReleaseNode(node_node);
```

www.axis.com
Graphical VX work flow

graph(ical) editor

Parser

OpenVX C code

Graph Manager

C library
Too many nodes!
Introducing subgraphs
Graph manager

Algorithm

- Graph1
- Graph2
- GraphN

Graph Manager

- register graphs
- chain subgraphs
- process graphs
- update I/O imgs

OpenVX

Graph1+2

input

input
The remaining code needed

```c
graphmanager_image_t input_images[] = { { src1, GRAPHMANAGER_IMG_TYPE_AXVX_UINT8 } };  
graphmanager_image_t output_images[] = { { output, GRAPHMANAGER_IMG_TYPE_AXVX_UINT8 } };  
io_param_t* io_params = io_param_create();  
io_params->input_images = input_images;  
io_params->output_images = output_images;

int graph_index = graphmanager_register_subgraph(graph_manager, graph_test, io_params, NULL, NULL);

graphmanager_set_active_subgraph(graph_manager, graph_index);  
graphmanager_process_graph(graph_manager);

graphmanager_set_active_subgraph(graph_manager, graph_index);  
input_images[0].image = src2;  
graphmanager_update_io_images(graph_manager, io_params);  
graphmanager_process_graph(graph_manager);
```
A graph to be parsed
A parsed graph with errors

ERROR: Incorrect naming of output edge labels (out2 and out2 might be.

ERROR: Node implementation not found. Maybe node name is wrong.
A parsed validated graph
Execution and development time comparison

<table>
<thead>
<tr>
<th>Development time (mw)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphical VX</td>
</tr>
<tr>
<td>OpenVX (C API)</td>
</tr>
<tr>
<td>Assembler</td>
</tr>
<tr>
<td>CPU implementation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Execution time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenVX</td>
</tr>
<tr>
<td>Assembler</td>
</tr>
<tr>
<td>CPU implementation</td>
</tr>
</tbody>
</table>

OpenVX implementation (portable performance)
Custom assembler implementation (performance)
CPU implementation (portable)
GitHub:

https://github.com/AxisCommunications/openvx-graph-parser
Thank you!