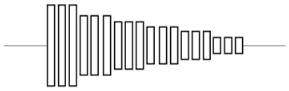
DX120P Single Chip Pose Decoder



Introduction

The DX120P is a Verilog-based CNN which maps a dense feature trajectory with shape [112,112,512] to a pose vector with shape [1,1,512] at 100 frames/s with 8.5ms latency. The DX120P has 8.4M weights and 14 layers as shown in Figure 1. Like the IE120R, the DX120P is a real time streaming component with AXI-S compatible interfaces. The Verilog runs at 250Mhz and can accept input rows at up to 12000 rows/s or 107 frames/s. The DX120P throughput and latency scale linearly with clock rate.

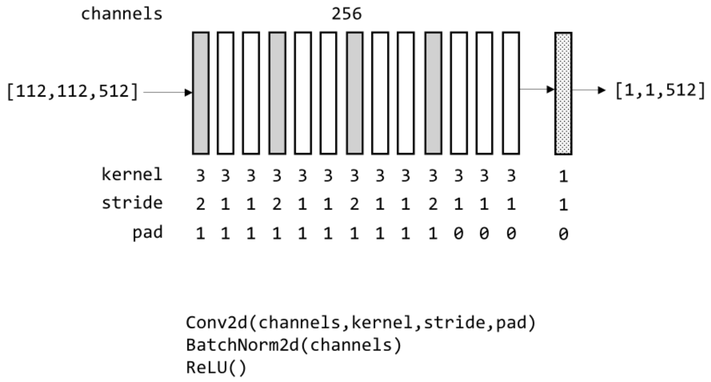

channels
256
[112,112,512]
[1,1,512]
kernel 3 3 3 3 3 3 3 3 3 3 3 3 3 1
stride 2 1 1 2 1 1 2 1 1 2 1 1 1 1
pad 1 1 1 1 1 1 1 1 1 1 0 0 0 0
Conv2d(channels,kernel,stride,pad)
BatchNorm2d(channels)
ReLU()

Figure 1

Verilog Module

The top-level Verilog module ports are the same as IE120R, except that both input and output are channel interleaved. Like the IE120R the TDMPAD parameters must be scaled to reflect the target input row rate. The default values are designed for an input rate of 12000 rows/s, or 107.1428571 frames/s.

```
module dx120p (
    input wire clk,
    input wire reset,
    input wire s_valid,
    input wire [$clog2(32):0] s_chan,
    input wire s_last,
    input wire [$clog2(112):0] s_col,
    input wire [$clog2(112):0] s_row,
    input wire [16*32-1:0] s_data,
    output wire m_valid,
    output wire [$clog2(64):0] m_chan,
    output wire m_last,
    output wire [$clog2(1):0] m_col,
    output wire [$clog2(1):0] m_row,
    output wire [8*32-1:0] m_data
);
```

Functional Verification

Functional verification uses the same method as IE120R to verify that the Verilog output matches the PyTorch activations.

```
devices/DX120P/verilog/sim
```

The waveform in Figure 2 shows the latency of 8.5ms from the last pixel in, to the last feature out, as well as continuous pipelined operation from frame to frame.

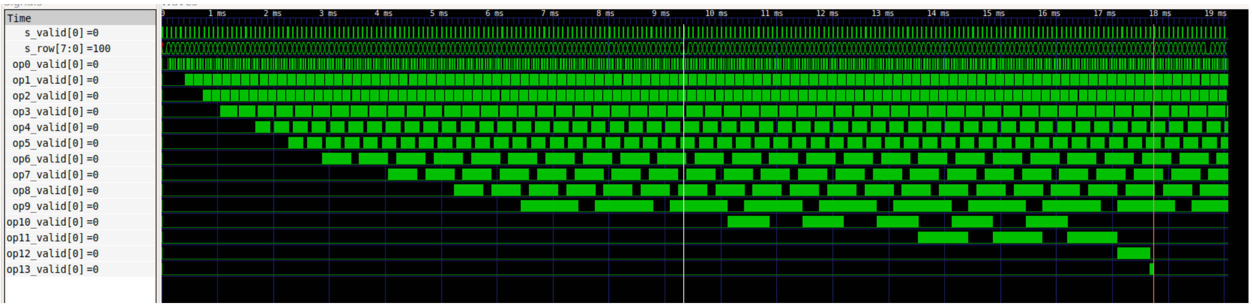


Figure 2

World Model Training

The DX120P comes untrained. It can be pretrained using a world model dataset which contains egocentric perception data with corresponding human imitation actions. The IE120R encoders can be pretrained and frozen, which reduces the end-to-end training problem to the DX120P decoder and LLM only. The LLM and DX120P pose decoder could also be pretrained and finetuned, either from scratch or using knowledge distillation from a frontier world model. TBD

```
pip install --upgrade siliconperception
```

```
import siliconperception
```

```
import torchinfo
```

```
print('siliconperception',siliconperception.__version__)
```

```
from siliconperception.DX120P import DX120P,DX120P_HW
decoder = DX120P()
torchinfo.summary(decoder,input_size=(1,512,112,112))
```

Quartus Compilation

The DX120P fits in the Agilex 027 device and has zero negative slack at 250MHz.

```
devices/DX120P/verilog/agilex
```

```
+-----------------------------------------------------------------------------+
; Flow Summary                                                                ;
+---------------------------------+-------------------------------------------+
; Flow Status                     ; Successful - Sat Jan  4 17:33:29 2025     ;
; Quartus Prime Version           ; 23.4.0 Build 79 11/22/2023 SC Pro Edition ;
; Revision Name                   ; top                                       ;
; Top-level Entity Name           ; top                                       ;
; Family                          ; Agilex 7                                  ;
; Device                          ; AGIB027R31B1I1V                           ;
; Timing Models                   ; Final                                     ;
; Power Models                    ; Preliminary                               ;
; Device Status                   ; Preliminary                               ;
; Logic utilization (in ALMs)     ; 257,332 / 912,800 ( 28 % )                ;
; Total dedicated logic registers ; 471393                                    ;
; Total pins                      ; 677 / 1,096 ( 62 % )                      ;
; Total block memory bits         ; 185,270,272 / 271,810,560 ( 68 % )        ;
; Total RAM Blocks                ; 11,741 / 13,272 ( 88 % )                  ;
; Total DSP Blocks                ; 4,560 / 8,528 ( 53 % )                    ;
; Total HSSI HPS                  ; 0 / 1 ( 0 % )                             ;
; Total PLLs                      ; 0 / 36 ( 0 % )                            ;
```

Reference Application

The IE120R and DX120P can be used to build an autonomous agile robot using the concept shown in Figure 3.

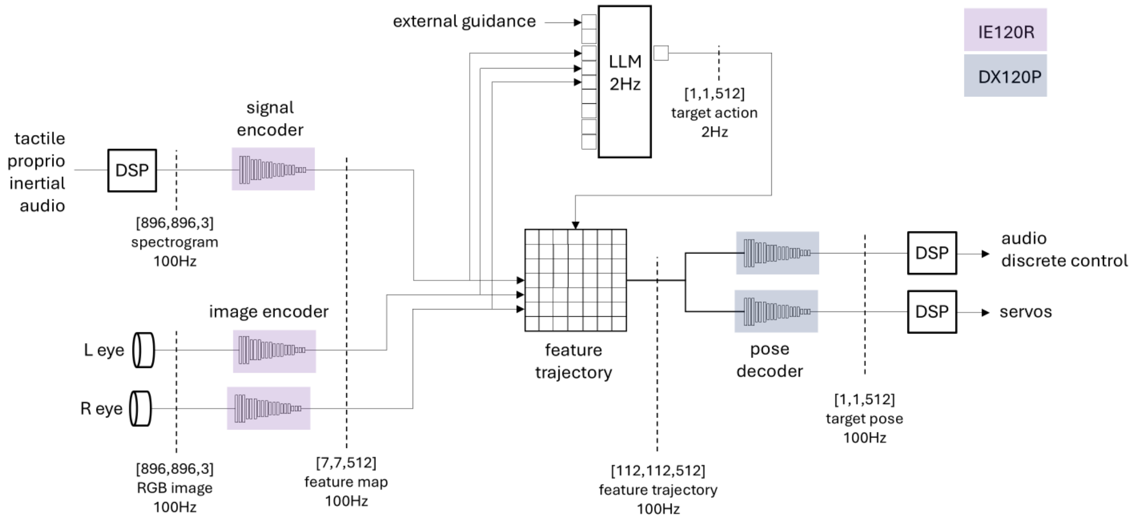


Figure 3

In this real time system, the raw sensor data is converted to feature maps using three IE120R devices (Agilex 027 silicon), a ~100X compression in bandwidth. The encoder weights can be pretrained and frozen.

The encoded feature maps are sent as a token stream to an LLM which provides 2Hz guidance to the pose decoder via the global target action with shape [1,1,512]. The perception tokens have shape [1,1,512] with rate 7*7*100*3=14700 tokens/s. Subsampling in time can be used to reduce the input token rate to the LLM. The LLM maps the input sequence to a target action which is stored in the feature trajectory to modulate the pose decoder.

The feature maps are stored in a circular sliding window to form a feature trajectory as shown in Figure 4. The ingress feature streams can be written into a flat circular buffer, while the output iterators reshape the features so that several space-time invariants are preserved. These include 1) the fixed 2D relationship between the 7x7 features within each feature map, and 2) fixed positions in the feature trajectory for time T0, T-1, T-2, ..., T-63, and 3) fixed relative positions for the left eye, right eye, signals, and global target action features. There are many ways to map the incoming feature streams into the feature trajectory, for example using space filling curves. The size of the feature trajectory storage is 112*112*512*4B = 25.7MB.

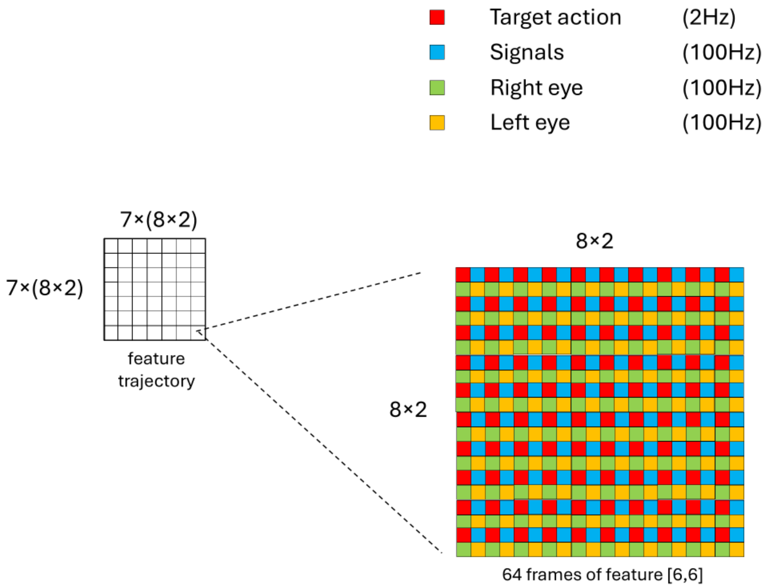


Figure 4

The feature trajectory is sent to the DX120P pose decoder sequentially, one row at a time, at 100 frames/s. The output of the pose decoder is a 100Hz stream of pose vectors with shape [1,1,512]. These pose vectors can be trained to directly predict servo angles or further decoded using DSP-based decompression.

In this concept, input signals are encoded by converting them into images using spectrograms or equivalent transformation. Output signals can be decompressed from the 51200 feature/s decoder output stream.