CUDAで動画再生

ぺりむ (@hi2p_perim)

利点

- CUDA GL interoperability reduces the bandwidth overhead
- Parallelism
- Much, much faster

Steps

repeat

- frame ← grab frame from the movie file
- wait for some time in order to do synchronization with movie fps
- draw frame to the screen

Grabbing frame

Some steps:

- Feeding compressed frame from video container (e.g. .avi, .mpg)
- Decode the frame according to the codec (MPEG-2, H.264/MPEG-4 AVC)
- Post processing the frame (e.g. color conversion)

NVCUVID

- Video decoding on GPU
- MPEG-2, VC-1, H.264 codec support

Traps

- Little documentation (only a pdf with 12 pages!)
- Requred some knowledge on CUDA driver API

CUDA runtime API and driver API

- Driver API is the lowest layer API
- Runtime API is constructed on the driver API
  - The famous syntax KernelCall<<<blocks, threads>>> is that of runtime API

CUDAでライブラリを作成する

- Driver APIを使う
- CUDAでライブラリを作るときの作法
  - 自分以外にも CUDA context が存在するかもしれない
  - cuCtxPopCurrent, cuCtxPushCurrent で他のCUDA context に作業できる余地を与える

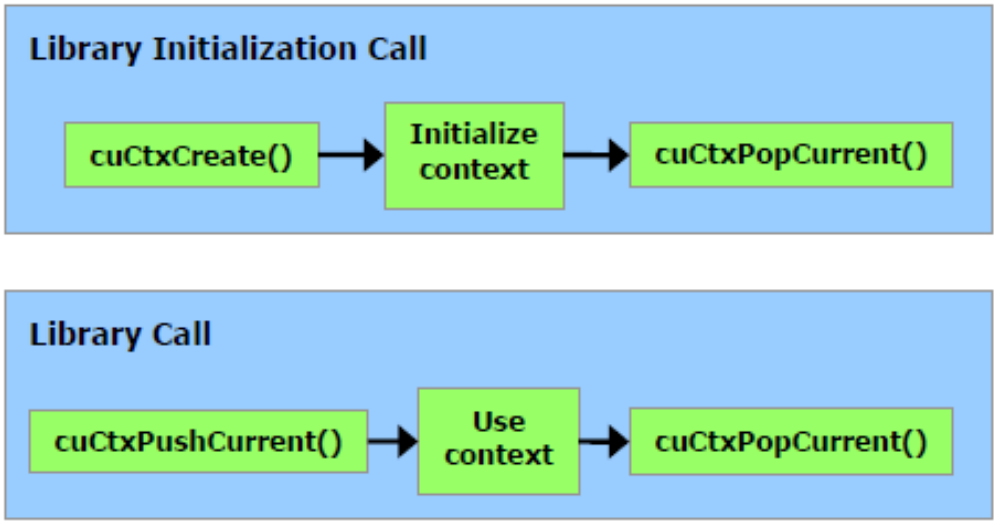

Figure G-1　Library Context Management

Implementation

Demo

- Run on NVIDIA GeForce GTX 285M
- Comparison with Quicktime
- Quicktime
  - ave. : around 25fps at 1080p
  - seems CPU-GPU data transmission is a bottleneck
- CUDA
  - ave. : around 80fps at 1080p

Implementation

Demo2

- Run on NVIDIA GeForce GTX 660 Ti
- Capable to run 24 QVGA movies simultaneously at 30fps (movie's original fps)