Formation OpenCV

Philippe FOUBERT

OpenCV (≥3.0) – Transparent API (UMat)

OpenCV – Transparent API

OCV 2.4: Face detect on CPU

```
// initialization
VideoCapture vcap(...);
CascadeClassifier
fd("haar_ff.xml");
Mat frame, frameGray;

vector<Rect> faces;

for(;;){
 vcap >> frame;
 cvtColor(frame, frameGray,
BGR2GRAY);
 equalizeHist(frameGray,
frameGray);

fd.detectMultiScale(frameGr
ay, faces);
}
```

OCV 2.4: Face detect using OpenCL™

```
// initialization
VideoCapture vcap(...);
ocl::OclCascadeClassifier
fd("haar_ff.xml");
ocl::oclMat frame, frameGray;
Mat frameCpu;
vector<Rect> faces;
for(;;){
 vcap >> frameCpu;
 frame = frameCpu;
 ocl:: cvtColor(frame, frameGray,
BGR2GRAY);
 ocl:: equalizeHist(frameGray,
frameGray);
 ocl::
fd.detectMultiScale(frameGray,
faces);
}
```

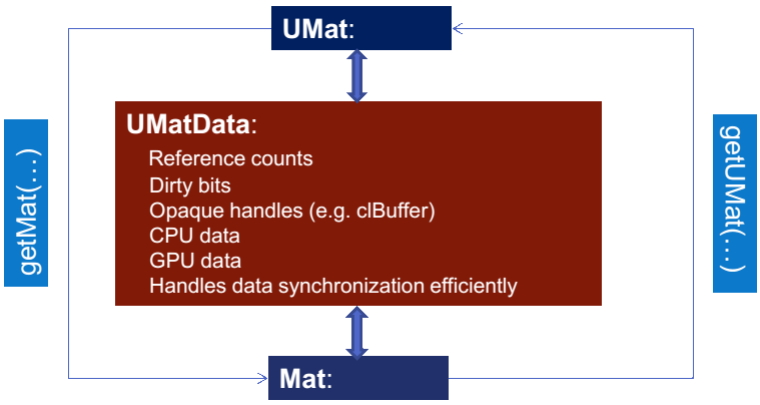
OCV 3.0: Face detect Anywhere!

```
// initialization
VideoCapture vcap(...);
CascadeClassifier
fd("haar_ff.xml");
UMat frame, frameGray;

vector<Rect> faces;
for(;;){
 vcap >> frame;


 cvtColor(frame, frameGray,
BGR2GRAY);
 equalizeHist(frameGray,
frameGray);

fd.detectMultiScale(frameGray
, faces);
}
```

OpenCV 2.4 - Similaire mais pas identique : il est nécessaire d'écrire explicitement du code CPU et du code OpenCL

Un code unique décliné différemment selon la plateforme

OpenCV – Transparent API - Principe



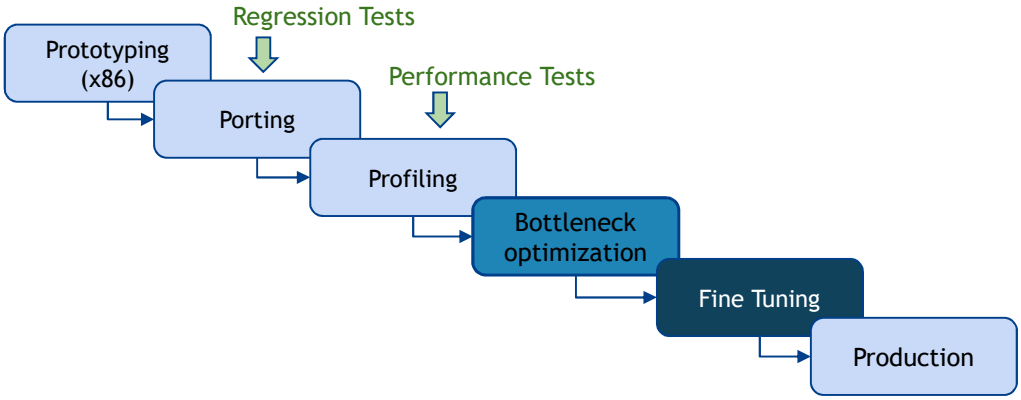
OpenCV – Transparent API – En interne

```
bool _ocl_cvtColor(InputArray src, OutputArray dst, int code)
{
  static ocl::ProgramSource oclsrc("//cvtcolor.cl source code …");
  UMat src_ocl = src.getUMat(), dst_ocl = dst.getUMat();
  if (code == COLOR_BGR2GRAY) {
     // get the kernel; kernel is compiled only once and cached
     ocl::Kernel kernel("bgr2gray", oclsrc, <compile_flags>);
     // pass 2 arrays to the kernel and run it
     return kernel.args(src, dst).run(0, 0, false);
  } else if(code == COLOR_BGR2YUV) { … }
  return false;
}

void _cpu_cvtColor(const Mat& src, Mat& dst, int code) { … }

// transparent API dispatcher function
void cvtColor(InputArray src, OutputArray dst, int code)
{
   dst.create(src.size(), …);
   if (useOpenCL(src, dst) && _ocl_cvtColor(src, dst, code)) return;
   // getMat() uses zero-copy if available; and with SVM it's no op
   Mat src_cpu = src.getMat();
   Mat dst_cpu = dst.getMat();
   _cpu_cvtColor(src_cpu, dst_cpu, code);
}
```


Processus de développement

Desktop for algorithm development

- **Video input, more debug possibilities, simple UI, higher speed**
- **Focus on algorithm, not environment**



Environnement