# 1. Mat

\* MatStep, the step of a line or multi-dimention-line

\* Mat::ptr<t>(int pos1, int pos2...), get the pointer of data at (pos1, pos2...) start from Mat::data

\* LUT

void cv::LUT( InputArray \_src, InputArray \_lut, OutputArray \_dst )

{

int cn = \_src.channels(), depth = \_src.depth();

int lutcn = \_lut.channels();

CV\_Assert( (lutcn == cn || lutcn == 1) &&

\_lut.total() == 256 && \_lut.isContinuous() &&

(depth == CV\_8U || depth == CV\_8S) );

CV\_OCL\_RUN(\_dst.isUMat() && \_src.dims() <= 2,

ocl\_LUT(\_src, \_lut, \_dst))

Mat src = \_src.getMat(), lut = \_lut.getMat();

\_dst.create(src.dims, src.size, CV\_MAKETYPE(\_lut.depth(), cn));

Mat dst = \_dst.getMat();

CV\_IPP\_RUN(\_src.dims() <= 2, ipp\_lut(src, lut, dst));

if (\_src.dims() <= 2)

{

bool ok = false;

Ptr<ParallelLoopBody> body;

if (body == NULL || ok == false)

{

ok = false;

ParallelLoopBody\* p = new LUTParallelBody(src, lut, dst, &ok);

body.reset(p);

}

if (body != NULL && ok)

{

Range all(0, dst.rows);

if (dst.total()>>18)

parallel\_for\_(all, \*body, (double)std::max((size\_t)1, dst.total()>>16));

else

(\*body)(all);

if (ok)

return;

}

}

LUTFunc func = lutTab[lut.depth()];

CV\_Assert( func != 0 );

const Mat\* arrays[] = {&src, &dst, 0};

uchar\* ptrs[2];

NAryMatIterator it(arrays, ptrs);

int len = (int)it.size;

for( size\_t i = 0; i < it.nplanes; i++, ++it )

func(ptrs[0], lut.ptr(), ptrs[1], len, cn, lutcn);

}

/\* example \*/

I = imread(...);

Mat lookUpTable(1, 256, CV\_8U);

uchar\* p = lookUpTable.ptr();

/\* set the table \*/

for( int i = 0; i < 256; ++i)

p[i] = table[i];

for (int i = 0; i < times; ++i)

LUT(I, lookUpTable, J);

# 2. MatOp

\* Matrix Operation, calculation

MatOp\_Solve, MatOp\_Initializer, MatOp\_Cmp … etc, in the MatOp.hpp and .cpp

# 3. LUT

class IppLUTParallelBody\_LUTCN : public ParallelLoopBody

The Parallel Run

In convert.cpp

ipp\_lut

In convert.cpp

class LUTParallelBody : public ParallelLoopBody

void cv::LUT( InputArray \_src, InputArray \_lut, OutputArray \_dst )