#ifndef IMAGE\_H

#define IMAGE\_H

#include "darknet.h"

#include <stdlib.h>

#include <stdio.h>

#include <float.h>

#include <string.h>

#include <math.h>

#include "image\_opencv.h"

#include "box.h"

#ifdef \_\_cplusplus

extern "C" {

#endif

/\*

typedef struct {

int w;

int h;

int c;

float \*data;

} image;

\*/

float get\_color(int c, int x, int max);

void flip\_image(image a);

void draw\_box(image a, int x1, int y1, int x2, int y2, float r, float g, float b);

void draw\_box\_width(image a, int x1, int y1, int x2, int y2, int w, float r, float g, float b);

void draw\_bbox(image a, box bbox, int w, float r, float g, float b);

void draw\_label(image a, int r, int c, image label, const float \*rgb);

void write\_label(image a, int r, int c, image \*characters, char \*string, float \*rgb);

void draw\_detections(image im, int num, float thresh, box \*boxes, float \*\*probs, char \*\*names, image \*\*labels, int classes);

void draw\_detections\_v3(image im, detection \*dets, int num, float thresh, char \*\*names, image \*\*alphabet, int classes, int ext\_output);

image image\_distance(image a, image b);

void scale\_image(image m, float s);

// image crop\_image(image im, int dx, int dy, int w, int h);

image random\_crop\_image(image im, int w, int h);

image random\_augment\_image(image im, float angle, float aspect, int low, int high, int size);

void random\_distort\_image(image im, float hue, float saturation, float exposure);

//LIB\_API image resize\_image(image im, int w, int h);

//LIB\_API void copy\_image\_from\_bytes(image im, char \*pdata);

void fill\_image(image m, float s);

void letterbox\_image\_into(image im, int w, int h, image boxed);

//LIB\_API image letterbox\_image(image im, int w, int h);

// image resize\_min(image im, int min);

image resize\_max(image im, int max);

void translate\_image(image m, float s);

void normalize\_image(image p);

image rotate\_image(image m, float rad);

void rotate\_image\_cw(image im, int times);

void embed\_image(image source, image dest, int dx, int dy);

void saturate\_image(image im, float sat);

void exposure\_image(image im, float sat);

void distort\_image(image im, float hue, float sat, float val);

void saturate\_exposure\_image(image im, float sat, float exposure);

void hsv\_to\_rgb(image im);

//LIB\_API void rgbgr\_image(image im);

void constrain\_image(image im);

void composite\_3d(char \*f1, char \*f2, char \*out, int delta);

int best\_3d\_shift\_r(image a, image b, int min, int max);

image grayscale\_image(image im);

image threshold\_image(image im, float thresh);

image collapse\_image\_layers(image source, int border);

image collapse\_images\_horz(image \*ims, int n);

image collapse\_images\_vert(image \*ims, int n);

void show\_image(image p, const char \*name);

void show\_image\_normalized(image im, const char \*name);

void save\_image\_png(image im, const char \*name);

void save\_image(image p, const char \*name);

void show\_images(image \*ims, int n, char \*window);

void show\_image\_layers(image p, char \*name);

void show\_image\_collapsed(image p, char \*name);

void print\_image(image m);

//LIB\_API image make\_image(int w, int h, int c);

image make\_random\_image(int w, int h, int c);

image make\_empty\_image(int w, int h, int c);

image float\_to\_image\_scaled(int w, int h, int c, float \*data);

image float\_to\_image(int w, int h, int c, float \*data);

image copy\_image(image p);

void copy\_image\_inplace(image src, image dst);

image load\_image(char \*filename, int w, int h, int c);

image load\_image\_stb\_resize(char \*filename, int w, int h, int c);

//LIB\_API image load\_image\_color(char \*filename, int w, int h);

image \*\*load\_alphabet();

//float get\_pixel(image m, int x, int y, int c);

//float get\_pixel\_extend(image m, int x, int y, int c);

//void set\_pixel(image m, int x, int y, int c, float val);

//void add\_pixel(image m, int x, int y, int c, float val);

float bilinear\_interpolate(image im, float x, float y, int c);

image get\_image\_layer(image m, int l);

//LIB\_API void free\_image(image m);

void test\_resize(char \*filename);

#ifdef \_\_cplusplus

}

#endif

#endif