## remap()

```
void cv::remap ( InputArray       src,
                 OutputArray      dst,
                 InputArray       map1,
                 InputArray       map2,
                 int              interpolation,
                 int              borderMode = BORDER_CONSTANT ,
                 const Scalar &   borderValue = Scalar()
               )
```

**Python:**

```
cv.remap( src, map1, map2, interpolation[, dst[, borderMode[, borderValue]]] ) -> dst
```

```
#include <opencv2/imgproc.hpp>
```

Applies a generic geometrical transformation to an image.

The function remap transforms the source image using the specified map:

$$\texttt{dst}(x,y) = \texttt{src}(map_x(x,y), map_y(x,y))$$

where values of pixels with non-integer coordinates are computed using one of available interpolation methods. $map_x$ and $map_y$ can be encoded as separate floating-point maps in $map_1$ and $map_2$ respectively, or interleaved floating-point maps of $(x,y)$ in $map_1$, or fixed-point maps created by using **convertMaps**. The reason you might want to convert from floating to fixed-point representations of a map is that they can yield much faster (2x) remapping operations. In the converted case, $map_1$ contains pairs (cvFloor(x), cvFloor(y)) and $map_2$ contains indices in a table of interpolation coefficients.

This function cannot operate in-place.

**Parameters**

- **src** Source image.
- **dst** Destination image. It has the same size as map1 and the same type as src .
- **map1** The first map of either (x,y) points or just x values having the type CV_16SC2 , CV_32FC1, or CV_32FC2. See **convertMaps** for details on converting a floating point representation to fixed-point for speed.
- **map2** The second map of y values having the type CV_16UC1, CV_32FC1, or none (empty map if map1 is (x,y) points), respectively.
- **interpolation** Interpolation method (see **InterpolationFlags**). The methods **INTER_AREA** and **INTER_LINEAR_EXACT** are not supported by this function.
- **borderMode** Pixel extrapolation method (see **BorderTypes**). When borderMode=**BORDER_TRANSPARENT**, it means that the pixels in the destination image that corresponds to the "outliers" in the source image are not modified by the function.
- **borderValue** Value used in case of a constant border. By default, it is 0.

**Note**

> Due to current implementation limitations the size of an input and output images should be less than 32767x32767.