

**NAME**

autopano – Generate panorama project from SIFT keypoint files.

**SYNOPSIS**

**autopano** [*options*] *output.pto* *keyfile1[.gz]* *keyfile2[.gz]* [*keyfile3[.gz]* [...]]

**DESCRIPTION**

Generate panorama project from keypoint data. First, read in two or more SIFT keypoint files, then correlate the keypoint data and build a list of matches. This list undergoes some filtering and the best matches are used to create control point information. The control point information are writing to a PTO compatible panorama project file. For example, programs such as **hugin** can read it.

Prior to any further use of the PTO, you ABSOLUTELY HAVE TO ADJUST THE FOCAL LENGTH of all the images. This can be done in hugin, in the "Camera and Lens" tab.

**OPTIONS**

--ransac <on|off|1|0>

Switch RANSAC postfiltration on or off. Default is on. There are only a few reasons to switch this off: if the keypoint density and matches are very sparse, RANSAC may filter too much. Or, if the lens geometry used is unusual (fish eye, very wide angle, micropanorama, ...) and does not resemble a rectilinear projection. That is, if you get really bad results with RANSAC on, disable it.

--maxmatches <count>

Set the maximum number of control point pairs you want to generate per image pair. The default is 16, so for each image pair, there are no more than 16 control points generated. If there are more control points to choose from, an area maximization metric is applied to keep the matches that cover most of the image area. You can disable this by setting count to zero. Then all matches are kept.

--disable-areafilter

At the final stage of creating control points, there is a list of matches for every image pair that overlaps. All this matches are thought to be correct and have been filtered using the RANSAC algorithm. However, often there are more matches available than the user wants to keep (see the "--maxmatches" option). In previous versions we applied a simple metric to pick out the matches that have a good matching score. Now, we have an area maximization algorithm, that maximizes the area covered by the matches. In general, this leads to better coverage of the image overlap area. However, if you want the old behaviour, that only considers the SIFT matching score of every match, enable this option.

--integer-coordinates

Use natural number coordinates in the PTO file for the found matches. Default is to use sub-pixel float coordinates to increase precision. You may want to try this option in case you use another frontend than hugin and you experience problems with the PTO files created by autopano-sift.

--absolute-pathnames <on|off|1|0>

Store the absolute pathnames of the image files in the PTO file. Normally, in case all images are in the same directory and the output PTO file is also saved in this directory, only the image file-names are used. Otherwise the absolute pathnames are used. Using this option you can enforce the behaviour.

*output.pto*

The filename of the PTO project file to generate.

*keyfile[.gz]*

The SIFT XML keypoint data file, as generated with the **generatekeys**(1) program. If the filename ends in ".gz", transparent **gzip**(1) decompression is used.

**ALIGN OPTIONS**

--align Enable fully automatic pre-aligning algorithm. This results in yaw, pitch and rotation coordinates being assigned to the images in the resulting PTO file. This option is not perfect yet, but in most

cases the result is far better than without using it.

There are a number of conditions on the input images that can be used with this algorithm. First, the images should all be of the same dimension, scale and have a simple (rectangular) geometry with roughly euclidean distances. Second, the order of the input images is considered so that the first images build an ordered row on the horizon. That is, the first, second, third, etc. images are strictly aligned left-to-right or right-to-left and all lie roughly on the horizon. They must also all be of the same rotation angle, which must be one of three rough cases: bottom-is-bottom, bottom-is-left, bottom-is-right. Bottom-is-top is forbidden. In case the bottom is either left or right, we estimate its position based on average keypoint density (also experimental).

In case the horizontal first row is not properly detected, try to increase the downscale resolution when creating the keypoint data. This will produce more keypoints which makes it easier to build the horizontal row. In case it still does not work, you should not enable this option. Please report bugs or successes with this option.

--bottom-is-left

--bottom-is-right

Only usable if --align option is enabled. If your input images are 90 or -90 degree tilted for the first row of horizontal images, you can force the orientation by telling the program where the bottom (floor) of the images is located: on the left or right side. If this option is not used, the program will attempt to automatically figure the orientation based on average keypoint density in the left and right half of the image.

--generate-horizon <count>

Generate a horizon from the first row of aligned horizontal images. This can only be used if the --align option is enabled. Then, up to count number of horizon control lines are written into the resulting PTO file at the middle of the first row images. The lines are optimally spaced and you should use values such as 2, 6 or 14 to get optimal results (the sum of power of two, starting with 2,  $2 + 4$ ,  $2 + 4 + 8$ ).

## REFINEMENT OPTIONS

--refine Enable the refinement step. The refining is done as last step before writing the PTO output file. For every matching control point pair a small patch in the original images is extracted at the original resolution. The image patches are matched against each other with the highest possible quality of matching, usually yielding dozens of keypoints. As this matches are derived from the original resolution image, their location is more precise and they are used to refine the original keypoint using one of the two methods below. Note that enabling the refinement step makes the total process longer, hence it is not enabled by default.

--refine-by-middle

--refine-by-mean

Two methods are available to choose the best point from the high resolution matches. Refine by middle searches the match closest to the original keypoints position and keeps only this match. Refine by mean builds the geometric center of all keypoints found in the patch and uses this coordinates.

--keep-unrefinable <on|off|1|0>

In case a good match cannot be refined because it is located too near to the boundary of the image, the original match is kept by default. To disable this behaviour, you can use this option, which throws away matches that cannot be refined. However, you might want to increase the number of matches to keep per image pair using the "--maxmatches" option then.

## COMPONENT IDENTIFICATION

The program complains about non-connected components in case there is one or more images or image groups which have no relationship to the rest of the images. This means there is no way to jump from an image in one group to another group by just following control point pairs between any two images. This

component identification is crucial for further optimization: if there is one or more non-connected components, global optimization based on control point pairs will be impossible.

To fix this problem, first identify the reason why there are no keypoint matches between the images in the different components. A common reason is that the images in one component are very diffuse and only have very few control points (such as images of the sky or water). If this is the case, you can try to increase the number of keypoints found in each image by increasing the `downscaleResolution` parameter of the **generatekeys(1)** program. Or you can add control point pairs between the images of the individual components manually, using software such as `hugin`. Another reason for different components could be that the images do not belong together to one panorama.

## COMMON PROBLEMS

PTO does not optimize/render in `hugin`. Please check that you adjusted the camera lens or focal length parameter for each image. We intentionally set it to a value that will disallow any operation from within `hugin` as to force the user to set this parameter. Without knowing this parameter, any work would be invalid, that's why we force such strict behaviour, sorry.

## BUGS

No bugs known, if you find any, please send a bug report to me. I will try to fix it.

## AUTHOR

Sebastian Nowozin <nowozin at cs dot tu dash berlin dot de>

## SEE ALSO

**autopano-sift(7)**, **autopano-complete(1)**, **generatekeys(1)**, **autopanog(1)**, **showone(1)**, **showtwo(1)**