# Summary

Both the EnhanceOCR rule object and task attempt to take existing OCR text and improve it by running image regions through one or more filters and comparing the OCR result on the filtered images to the original OCR results. All incarnations use the original document text (from uss file) and will fail if the uss is not present.

Enhance OCR task: Processes the entire area of every page. The uss file is updated with the results.

Enhance OCR PP: Processes the entire overall bounds of the input document text on each page. The input document text is discarded and replaced with the results (the uss file is not affected).

Enhance OCR VM/OH: Processes the each raster zone of the supplied text separately. The value of the attribute is discarded and replaced with the result. (Behaves much the same as Re-OCR image area except better results + slower)

If more than one attribute is likely on a page, it is more efficient to process the zones via an OH rather than individually via the VM.

# Settings

* **OCR confidence threshold:** Any text not already existing in the file’s OCR output or words that contain characters below this threshold will be “enhanced” if possible. The higher the setting the slower the processing and, up to a point, the better the results. That point tends to be around 60% (see table 1 below).
* **General enhancement slider:** Specifies the number of possible filters that may be run in an attempt to enhance the OCR text. (See filter section below for specifics). The higher the setting the slower the processing and better the results. (see table 2 below)
* **Halftone or speckled radio button:** Similar speed to “Medium” slider option, but better results on text in halftone or speckled regions.
* **Aliased of diffuse radio button:** Similar speed to “Medium” slider option, but better results on aliased or diffuse text.
* **Smudges or lines radio button:** Similar speed to “Medium” slider option, but better results on text with lines or smudges.
* **Custom radio button:** Allows a custom-defined filter set to be used in place of a built-in version.
* **Preferred format regex:** When evaluating different OCR output, output that that contains a match for the specified format will be strongly preferred over one that does not.
* **Characters to exclude:** Any characters specified here will not be added by the enhancement process. Specified characters already existing in the OCR text will not necessarily be removed.
* **Output filtered images (debug):** In order to evaluate the effect of the filters used during enhancement, a filtered image for each page and filter will be output alongside the original image. Note that if an EnhanceOCR instance is called more than once per ruleset, each successive execution will overwrite the previous filter output so that upon completion, only the output from the final execution will be present.

**Algorithm**

The general algorithm used for each document page is:

\* Indicates steps that are used by the task only and PP only.

1. \* Remove any black borders on the page.
2. \* Erase all words from the document where no character is below the OCR confidence threshold
3. Apply filter “medium-45” (explained below) which removes most noise.
4. \* Regions of remaining pixel content on the page are identified for enhancement (SRICA with lines excluded)
5. For each filter (starting with filter “medium-45”):
   1. The filtered page is OCR’d
   2. The result is stripped of any characters in the chars to exclude field.
   3. The OCR’d text for each zone is compared to the original zone’s text with one of these 3 results:
      1. If the OCR confidence is substantially better than the existing result, the existing result zone text is completely replaced with the OCR text.
      2. If the OCR confidence is substantially worse than the existing result, it is disregarded.
      3. If the OCR confidence is similar to the existing text, the text is merged on a character by character basis based on several factors but mostly confidence.
   4. Each zone where all characters now meet the confidence threshold are removed from the list of zones to enhance.
   5. A new filtered image is generated with just the zones to enhance where the original text has been run through the next filter.
6. If a preferred format regex is used, if one or more OCR results (from 5c) or the original zone text contains a match for the regex pattern, the one with the highest confidence replaces the existing result unless the existing result has a higher confidence by a very large margin.
7. The result for each zone is applied back to the original document text.

# Filters

Each pass in step 5 of the algorithm above will use a different filter process on the original image text. Each pass will use either a single filter on its own or a combination of 2 separate filters. The filter can be one of a number of built-in filters available in the LeadTools API, a built-in Extract -developed filter, or a custom filter defined in a file available to the processing machine. Each filter will have zero, one or more parameters that can be adjusted, the most common being a setting defining the threshold (or bias) at which a pixel should become at least 50% saturated (resulting in an output color of black).

Note that though Extract software deals primarily in 2-color black-and-white images, for most filters the images will be processed as 16-bit images since this is required for many LeadTools filters and allows for combining the results of 2 different filters in one pass.

**Medium**: A 7x7 derivative of a Gaussian filter where all pixels x pixels from the target pixel are weighted the same as those x + 1 pixels from the center. The filter can either remove noise or fill-in/saturate existing pixels. The single parameter specifies the percentage at which the target pixel becomes 50% saturated. For example: 45 means that the weighted pixel value must be at least 45% of the max possible value (in image filter lingo, this is the same as having a bias of -0.05). Higher values remove noise while lower values saturate existing pixel content.

0, 0, 3, 3, 3, 0, 0,

0, 3, 4, 4, 4, 3, 0,

3, 4, 6, 6, 6, 4, 3,

3, 4, 6, 48, 6, 4, 3,

3, 4, 6, 6, 6, 4, 3,

0, 3, 4, 4, 4, 3, 0,

0, 0, 3, 3, 3, 0, 0

**Large**: A 9x9 version of the medium filter.

0, 0, 1, 1, 1, 1, 1, 0, 0,

0, 1, 1, 2, 2, 2, 1, 1, 0,

1, 1, 2, 3, 3, 3, 2, 1, 1,

1, 2, 3, 5, 5, 5, 3, 2, 1,

1, 2, 3, 5, 36, 5, 3, 2, 1,

1, 2, 3, 5, 5, 5, 3, 2, 1,

1, 1, 2, 3, 3, 3, 2, 1, 1,

0, 1, 1, 2, 2, 2, 1, 1, 0,

0, 0, 1, 1, 1, 1, 1, 0, 0

**Small**: A 5x5 version of the medium filter.

3, 4, 4, 4, 3,

4, 6, 6, 6, 4,

4, 6, 48, 6, 4,

4, 6, 6, 6, 4,

3, 4, 4, 4, 3

**Despeckle**: A built-in LeadTools filter that removes speckles from a bitmap. This filter takes no parameters. NOTE: This appears to be identical to the Median filter with a parameter of 3.

**Average**: A built-in LeadTools filter that changes the color of each pixel in a bitmap to the average color of pixels in its neighborhood. This results in a blur effect. The filter takes a single parameter that specifies the size of the averaging neighborhood. It is the length, in pixels, of one side of a square.

**Median**: A built-in LeadTools filter that changes the color of each pixel in a bitmap to the median color of pixels in its neighborhood. This is similar to the average filter, but it is used for noise reduction, rather than a blur effect. The filter takes a single parameter that specifies the dimensions of the neighborhood used for filtering (dim x dim), in pixels.

**Gaussian**: A built-in LeadTools filter that changes the color of each pixel in a bitmap to the median color of pixels in its neighborhood. This is similar to the average filter, but it is used for noise reduction, rather than a blur effect. The filter takes a single parameter that specifies the dimensions of the neighborhood used for filtering (dim x dim), in pixels. NOTE: This filter runs very slowly.

**Sharpen**: A built-in LeadTools filter that increases or decreases the sharpness of the image in the specified bitmap. The filter takes a single parameter from -100 to 100 where positive values increase sharpness and negative values decrease sharpness.

**Smooth**: A built-in LeadTools filter that smooths the bumps and fills in the nicks of a 1-bit black and white image. The filter takes two parameters: The first specifies the size of the bump (or nick) to remove (or fill). All bumps and nicks less than or equal to this size are processed. This value is in pixels. The second is flags that can be used in combination though only one flag is currently available and meaningful:  
 256: Processes long bumps or nicks before short bumps or nicks. If this flag is not passed, short is favored over long.

**Min**: A built-in LeadTools filter that dilates dark objects by the specified amount. The single parameter specifies the dimensions of the neighborhood used for filtering (dim x dim), in pixels. The effect can be controlled by specifying the size of the neighborhood that is used for calculating each pixel value. For example, for 8x8, pass 8 in the dim parameter. Each pixel is replaced with the minimum value of its neighborhood.

**Max**: A built-in LeadTools filter that erodes dark objects by the specified amount. The single parameter specifies the dimensions of the neighborhood used for filtering (dim x dim), in pixels. The effect can be controlled by specifying the size of the neighborhood that is used for calculating each pixel value. For example, for 8x8, pass 8 in the dim parameter. Each pixel is replaced with the maximum value of its neighborhood.

**HighPass**: A built-in LeadTools filter that removes low frequency details in a bitmap, resulting in a sharpened image. The filter takes two parameters: The first indicates the radius of the neighborhood on which the filter is applied, in pixels. The second represents the intensity of the effect. The valid range is from 0 to 100. At 100, only the object outlines remain.

**Original**: No filter is applied. This option simply allows OCR another pass in a different context.

# Filter sequences

The built-in filter sequences are as follows where a dash char (-) delimits parameters and a plus char (+) delimits separate filters to be combined:

**Low**

medium-45 (versatile)

**Medium**

medium-45 (versatile)

despeckle (for speckles)

small-20 (aliased or diffuse text)

medium-85 (heavily shaded regions)

**High**

medium-45 (versatile)

medium-85 (heavily shaded regions)

medium-15 (for very diffuse/broken text)

despeckle (versatile)

small-20 (aliased or diffuse text)

NOTE: Though the filters medium-85 or medium-15 are generally not likely to produce good OCR, for the zones they do improve, best results are obtained by running them before most others since running other filters on these zones first sometimes produces incorrect results that have high enough confidence to prevent the correct result from being used later on. On the other hand these two filters rarely produce OCR results for zones that aren't helped by these filters so they tend to do little harm by being run earlier in the sequence-- the biggest harm is slower performance since these two filters end up being run for most zones while the help very few.

**Halftone/Speckled**

medium-45 (versatile)

medium-85 (heavily shaded regions)

despeckle (for speckles)

medium-55 (for somewhat heavily shaded regions)

**Aliased/Diffuse**

small-20 (aliased or diffuse text)

gaussian-1+medium-15 (aliased or diffuse text)

medium-40 (versatile)

small-25 (aliased or diffuse text)

**Lines/Smudges**

small-60 (smudged text or text with lines)

gaussian-1 (smudged text or text with lines)

medium-55+large-45 (smudged text or text with lines)

gaussian-4 (smudged text or text with lines)

# Custom Filters

Custom filters and filter sequences can be supplied via encrypted dat (text) files. The file referenced directly from the task/rule configuration defines the filter sequence to use and should be formatted with each pass on a separate line in the file as in the above section (without comments in parenthesis, although comments following double slashes // are supported). Built-in filters can be specified or a filter can be referenced that is defined in a dat file parallel to the sequence file where the filename matches the filter name used. Note that if filter definition file exists under the same name as an existing filter, the custom filter definition file will be used in place of the built-in filter.

A filter definition file should contain a series of integers delimited by commas where the total number of integers is a perfect square. Newlines are technically ignored, but practically speaking it is best to format the file so that each row in the file corresponds to a row of pixels where the pixel being acted upon is represented by the middle number (as the small, medium and large filters are defined above).

Example:

**customSequence.dat.etf**

happy-50

despeckle

medium-85

**happy.dat.etf**

0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0,

0, 0, 8, 0, 8, 0, 0,

0, 0, 0, 0, 0, 0, 0,

0, 8, 0, 0, 0, 8, 0,

0, 0, 8, 8, 8, 0, 0,

0, 0, 0, 0, 0, 0, 0