Supplementary Information

Detection and Segmentation of Cell Nuclei in Virtual Microscopy Images: A Minimum-Model Approach

Stephan Wienert[1,2], Daniel Heim[2], Kai Saeger[2], Albrecht Stenzinger[3], Michael Beil[4], Peter Hufnagl[1], Manfred Dietel[1], Carsten Denkert[1], Frederick Klauschen[1*]

[1] Institute of Pathology, Charité University Hospital Berlin, Charitéplatz 1, 10117 Berlin, Germany

[2] VMscope GmbH, Charitéplatz 1, 10117 Berlin, Germany

[3] Institute of Pathology, University Hospital Heidelberg, Im Neuenheimer Feld 220/221, 69120 Heidelberg, Germany

[4] Department of Medicine I, University of Ulm, Albert-Einstein-Allee 23, 89081 Ulm, Germany

* Correspondance should be addressed to frederick.klauschen@charite.de

Contents

Note S1: Method validation

To assess the accuracy of the proposed method 7931 cells from 36 images were labeled by three pathologists (FK, AS, CD). All images were labeled by three pathologists and only cells for which consensus was achieved were included. All other cells were left unlabeled. A comparison to Al-Kofahis method (Al-Kofahi et al. 2010) was performed using first the default (fully automatic) mode (which was recommended by the authors upon request) and second, using an optimized parameter configuration file provided by the authors after submitting test data to them. The images (600x600 pixels, Hematoxylin-Eosin stained) were taken from previously digitized routine diagnostic cases (see Methods for further details). After manual labeling the images were segmented and true positive (tp), false negative (fn) and false positive (fp) events were automatically determined using the positional information previously obtained in the manual "gold-standard" cell labeling. Cells touching the image border were excluded from the analysis. Image pairs show 1) the original image and 2) the image showing the nucleus contours (green) combined with the manually assigned labels that were automatically classified as true positive (green dots), false negative (red dots) and false positive (yellow dots) according to the following definition:1) true positive: cell nuclei whose area contained a pathologist-assigned label. A contour containing two pathologist-assigned labels will result in two true positives; 2) false positive: objects defined as cell nuclei by automated segmentation but with no manual label; 3) false negative: manually labeled cell nuclei not detected in the segmentation result. A conglomerate score was calculated to evaluate the ability of the algorithms to correctly separate conglomerates. The computation time of each image was also measured. All algorithms used were parallelized. The time measured includes the time executing the color deconvolution for the algorithm presented in this paper but it is not included in the time of the Al-Kofahi algorithms. On the other hand the Al-Kofahi algorithms were started as extra process so some process overhead time is included in their times.

Statistical values are defined as follows:

- Precision = tp / (tp + fp).
- Recall = tp / (tp + fn).
- Conglomerate=(Number of detected cells-fp)/tp

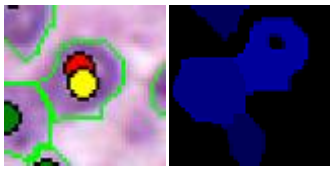
All other ratios are computed dividing the count of samples specified in the respective

column by the nuclei count.

The following pictures always are ordered as follows: original, result of “Wienert”, result of “Al-Kofahi quality”. The results of “Al-Kofahi speed” are not visualized but listed in the tables.

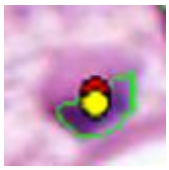
The result images contain different scenarios where the result may be hard to understand and therefore shall be described here briefly.

Contour containing red and yellow dots:



This can happen because only outer contours are rendered. The map of this image shows a hole in this cell core, so the pixel in the map where the pathologist clicked to mark the nuclei is not marked in the result label map of the algorithm.

Contour seems to fit nuclei but is false positive with a close false negative point:

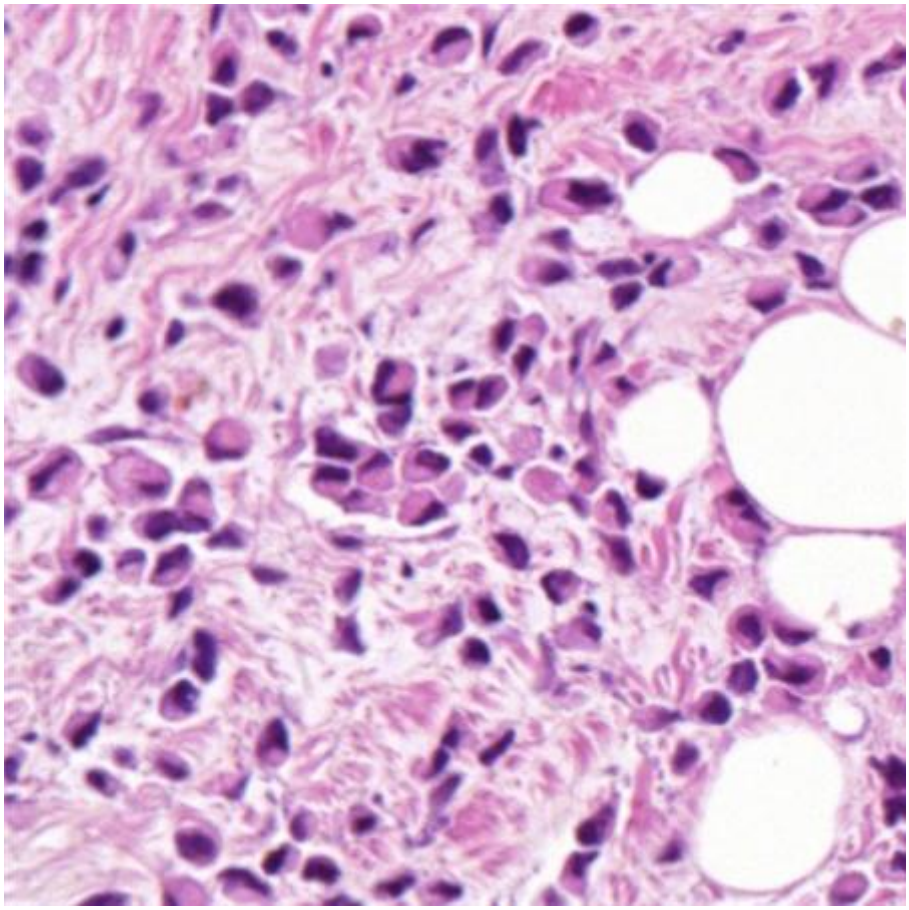


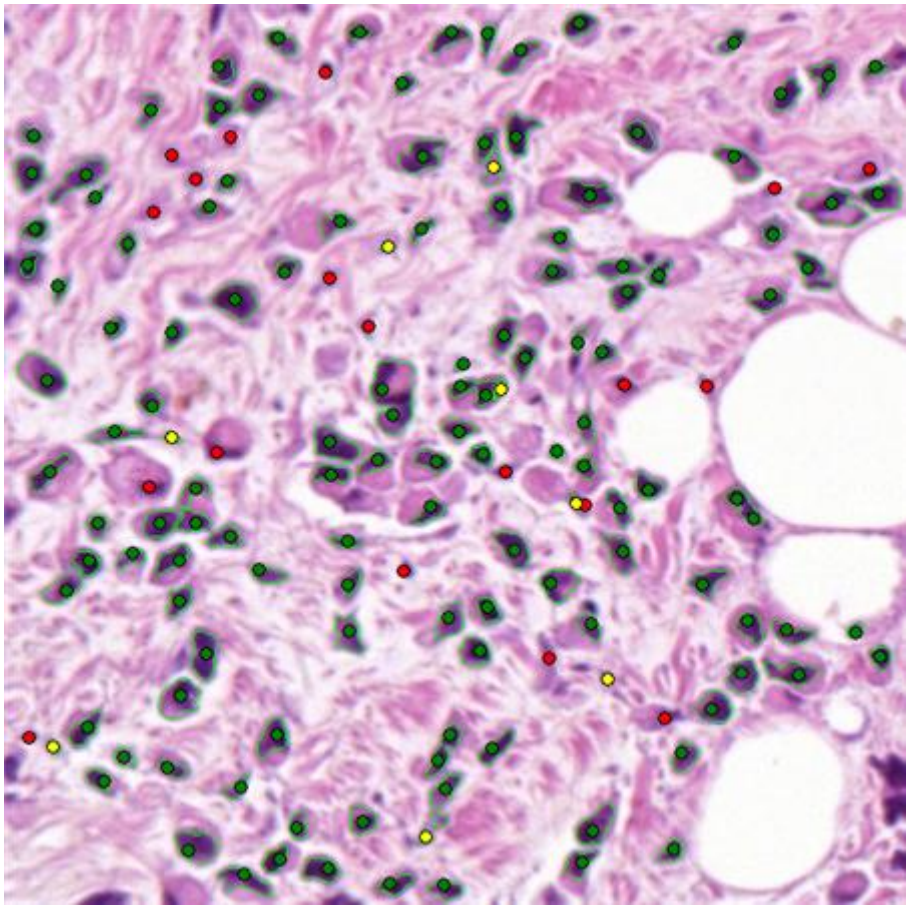
This is because only one pixel was labeled by the pathologist and the contour does not contain or enclose that point.

False negatives or false positives that you would not expect at the image border:

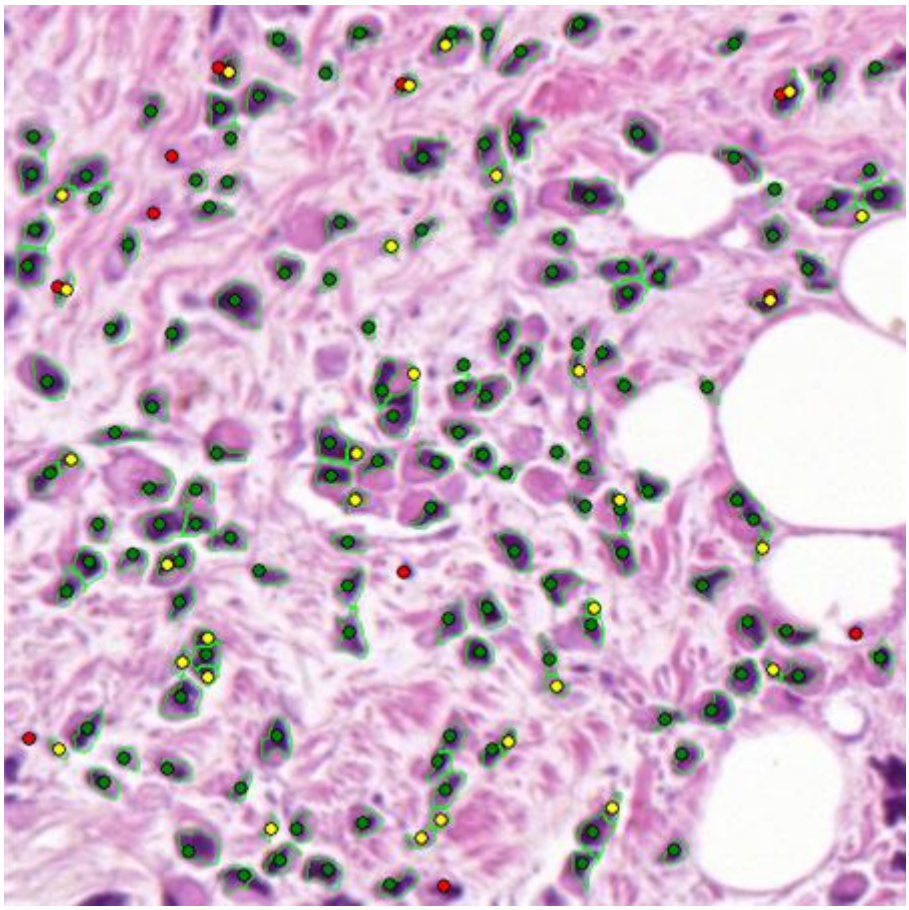
Contours with pixels at the image border were removed and only nuclei completely contained by an image were labeled. So a too wide contour at the image border may result in false negative and a too small contour in false positive.

1





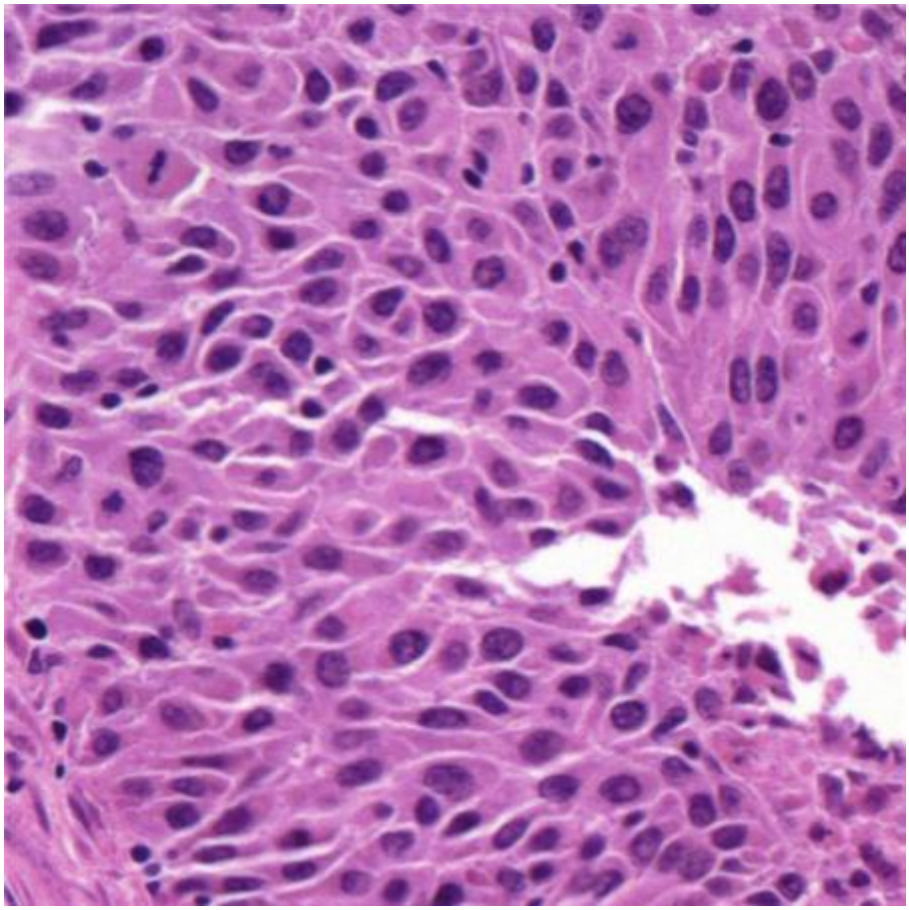
Wienert

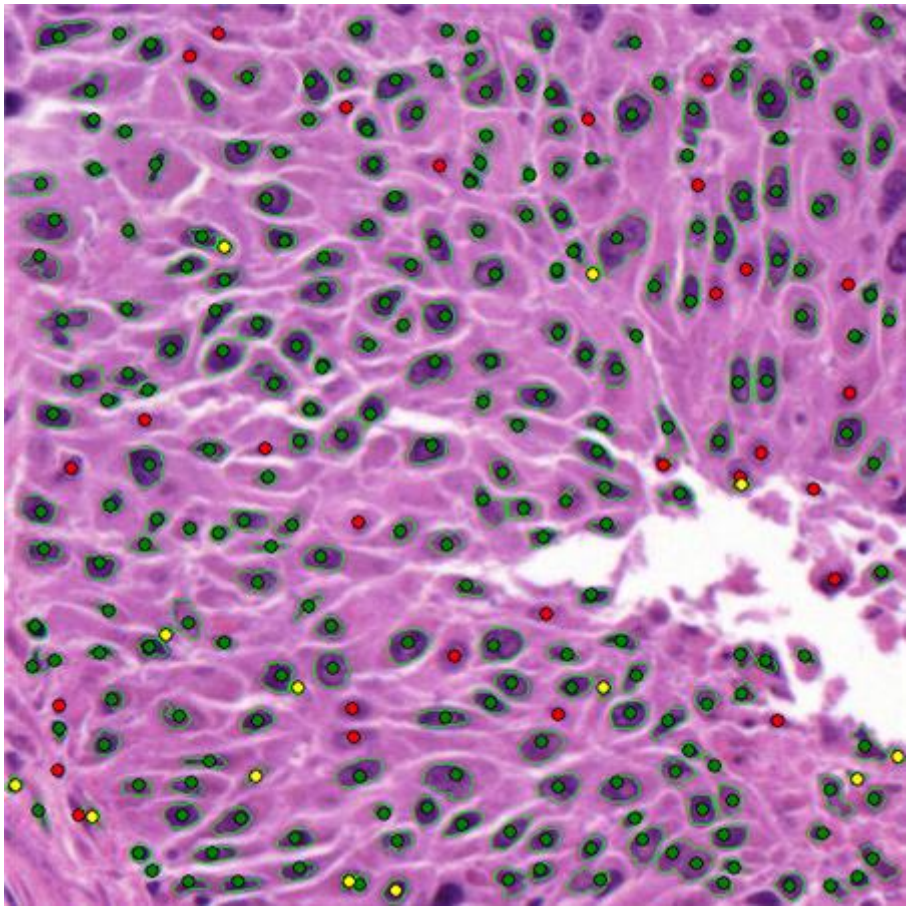


Al-Kofahi (optimized)

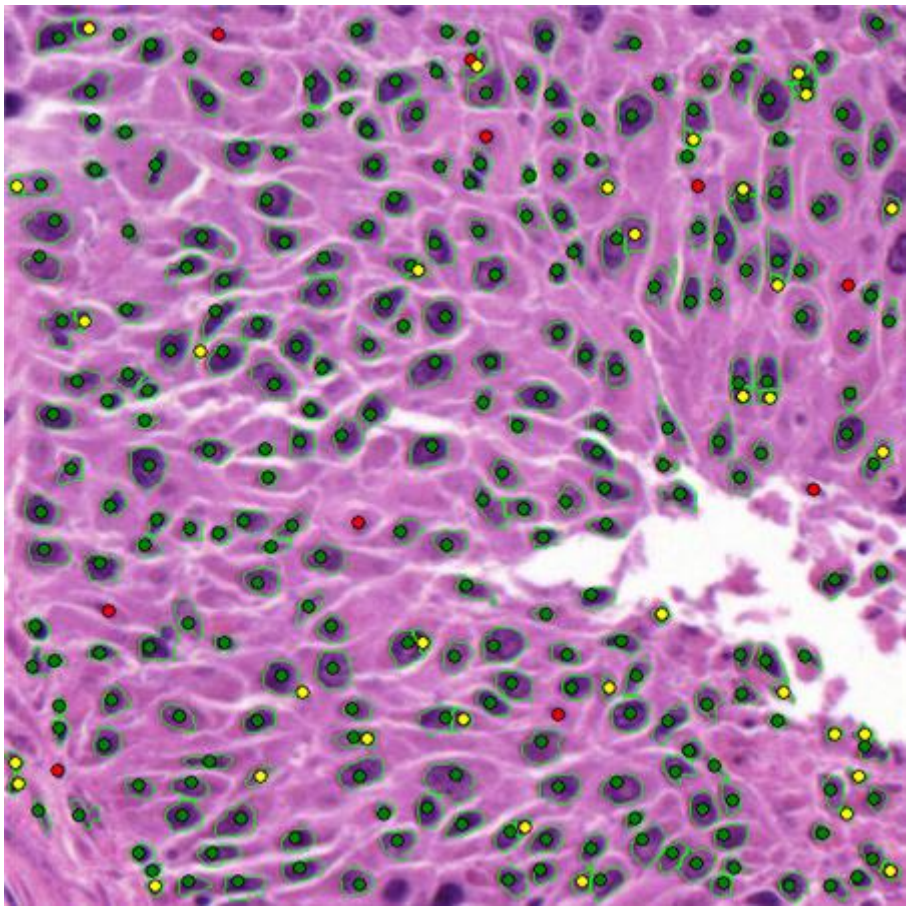
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	147	147	147
true positive count	128	136	136
false negative count	19	11	11
false positive count	8	36	30
precision	0,941	0,791	0,819
recall	0,871	0,925	0,925
conglomerate	0,969	0,993	0,993
time(ms)	702	624	1685

2





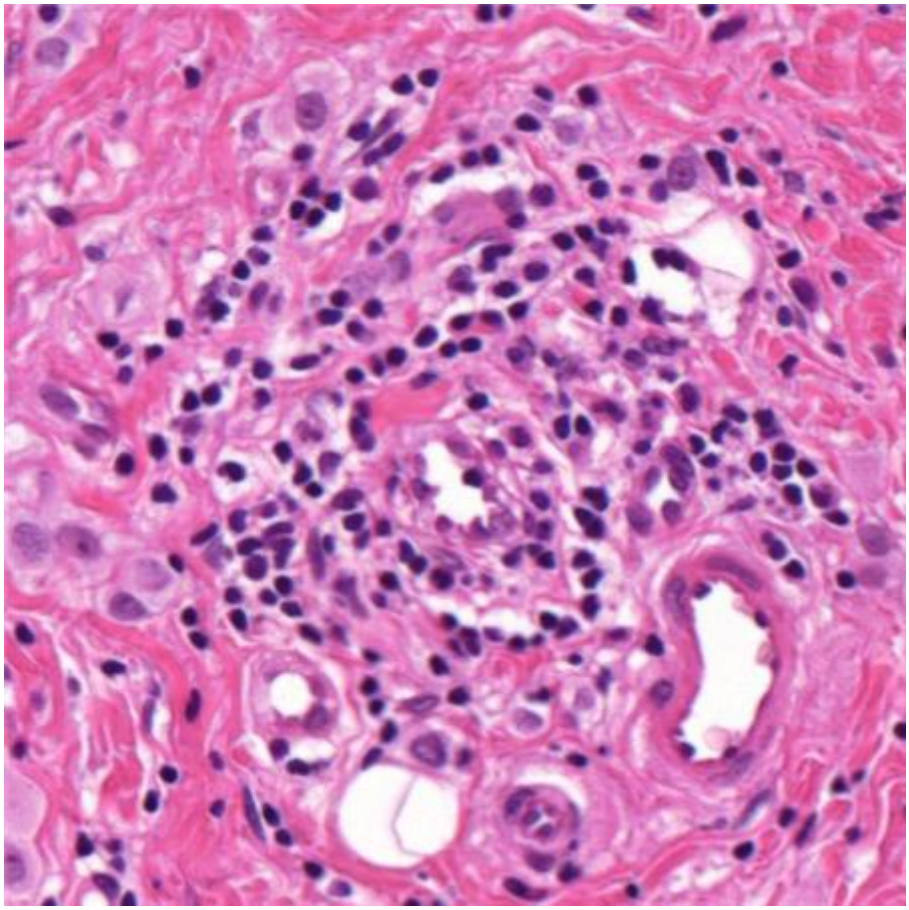
Wienert

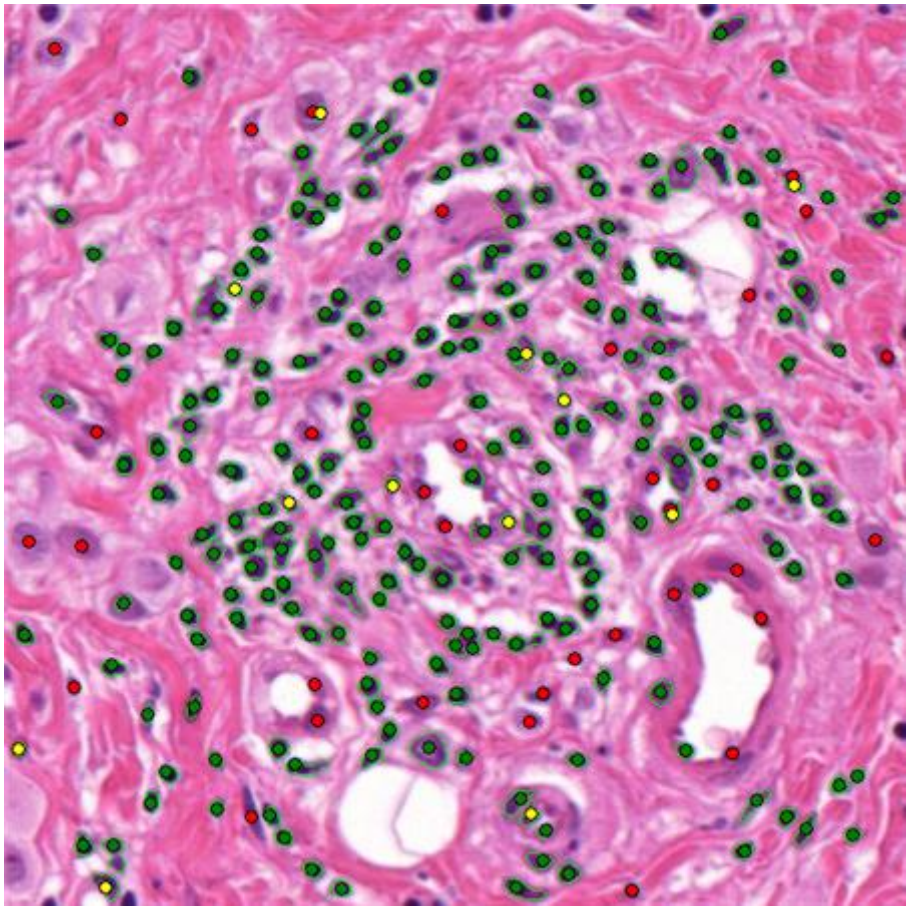


Al-Kofahi (optimized)

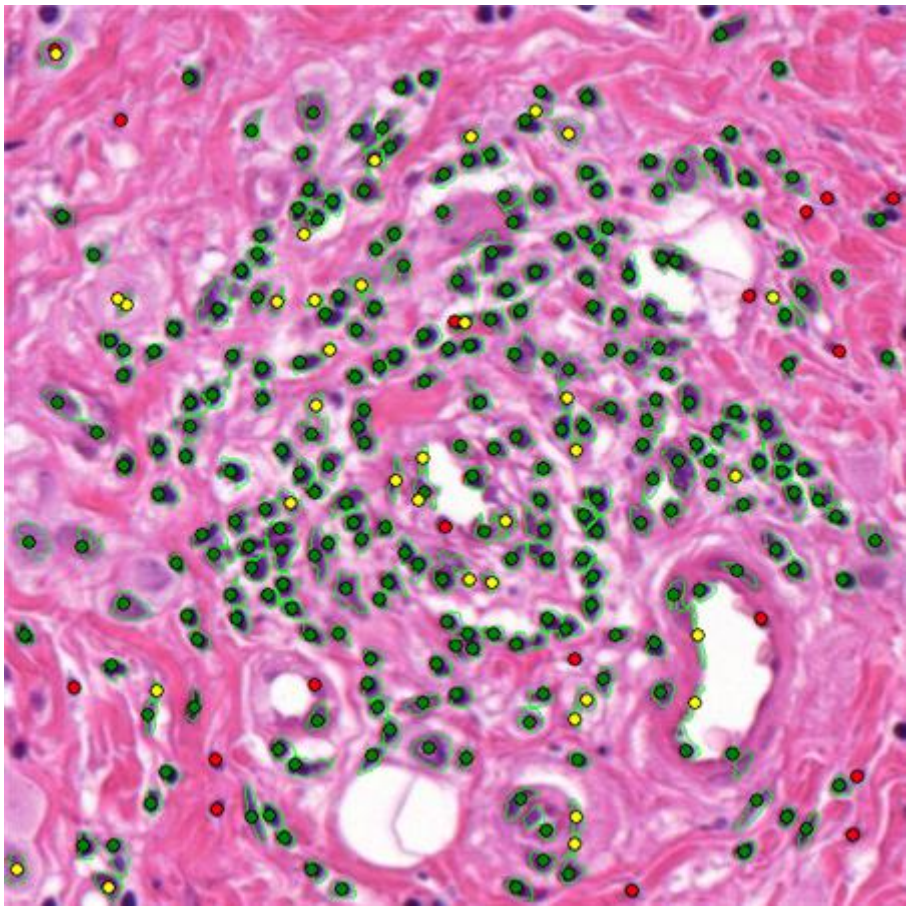
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	265	265	265
true positive count	236	254	255
false negative count	29	11	10
false positive count	13	43	35
precision	0,948	0,855	0,879
recall	0,891	0,958	0,962
conglomerate	0,992	0,988	0,988
time(ms)	764	639	2137

3





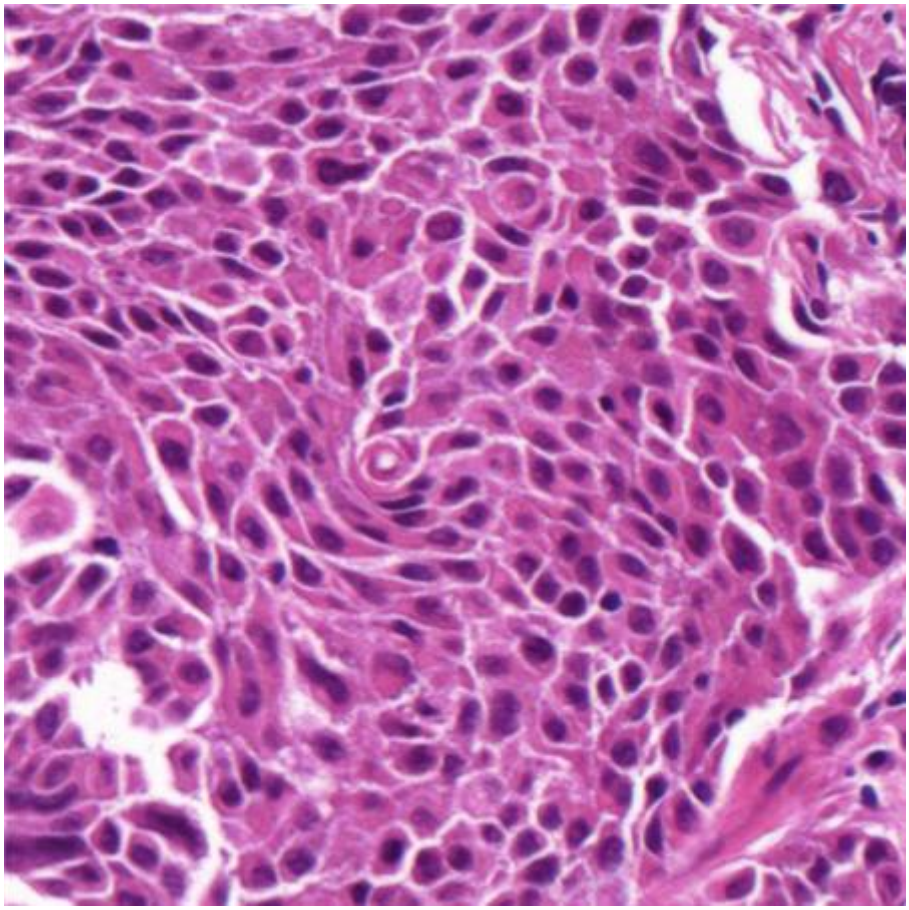
Wienert

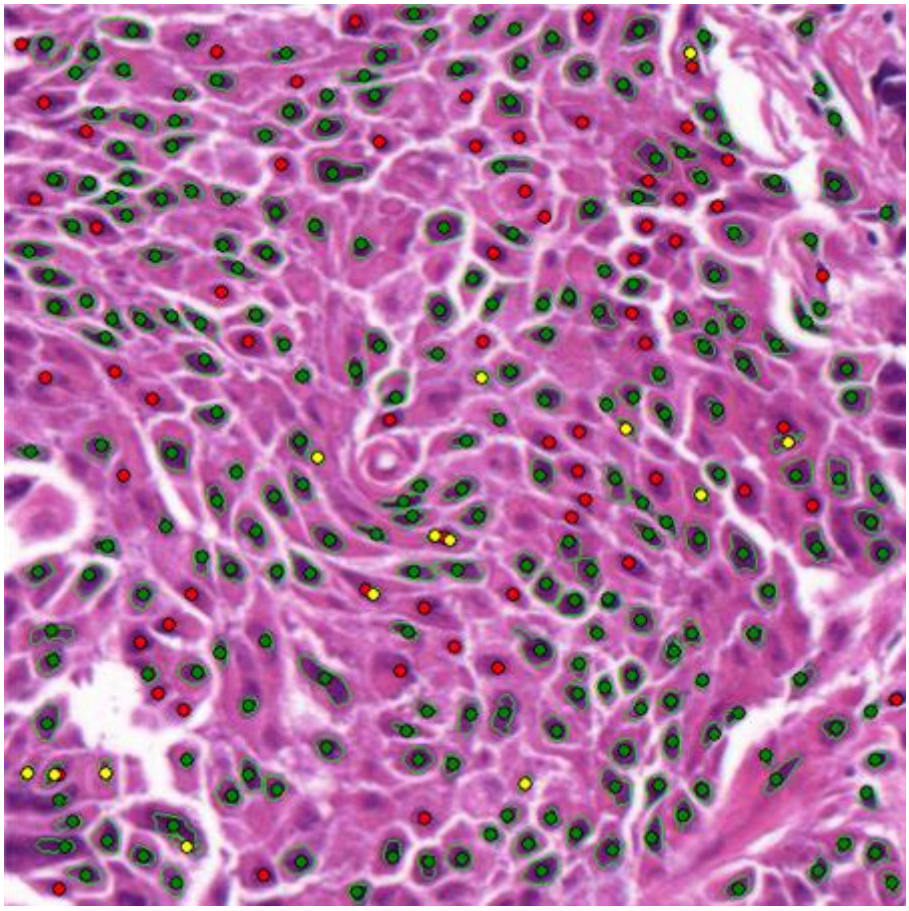


Al-Kofahi (optimized)

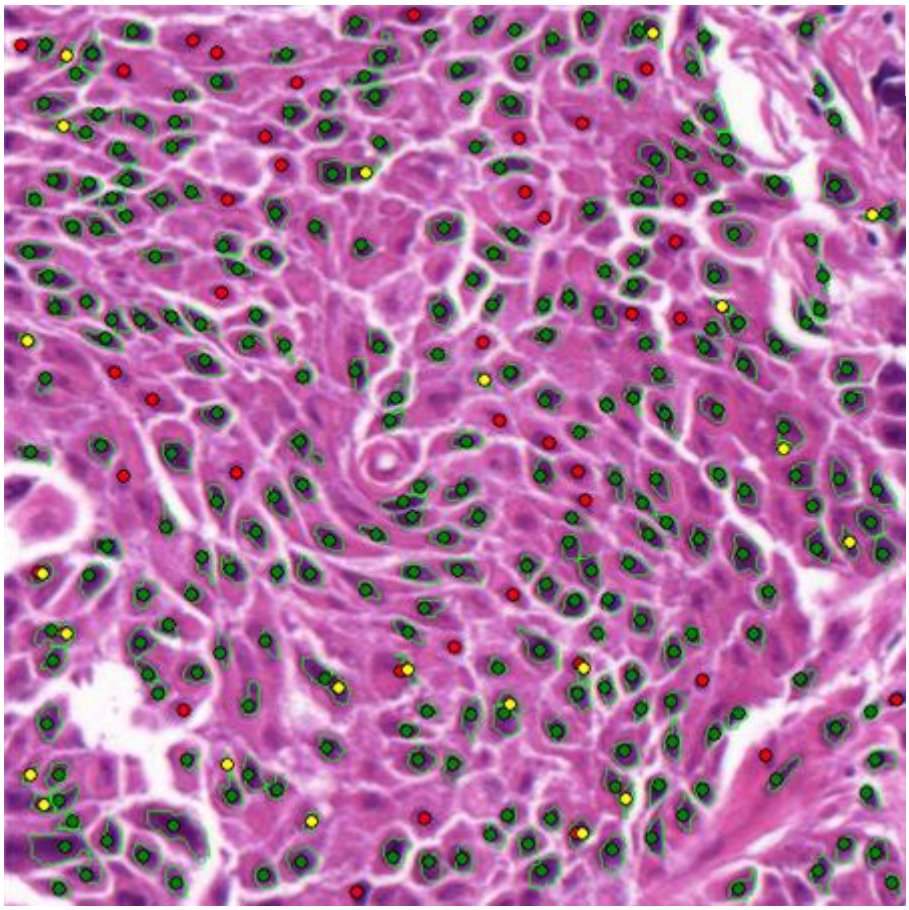
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	270	270	270
true positive count	235	252	252
false negative count	35	18	18
false positive count	12	96	34
precision	0,951	0,724	0,881
recall	0,870	0,933	0,933
conglomerate	0,906	0,948	0,956
time(ms)	686	655	1950

4





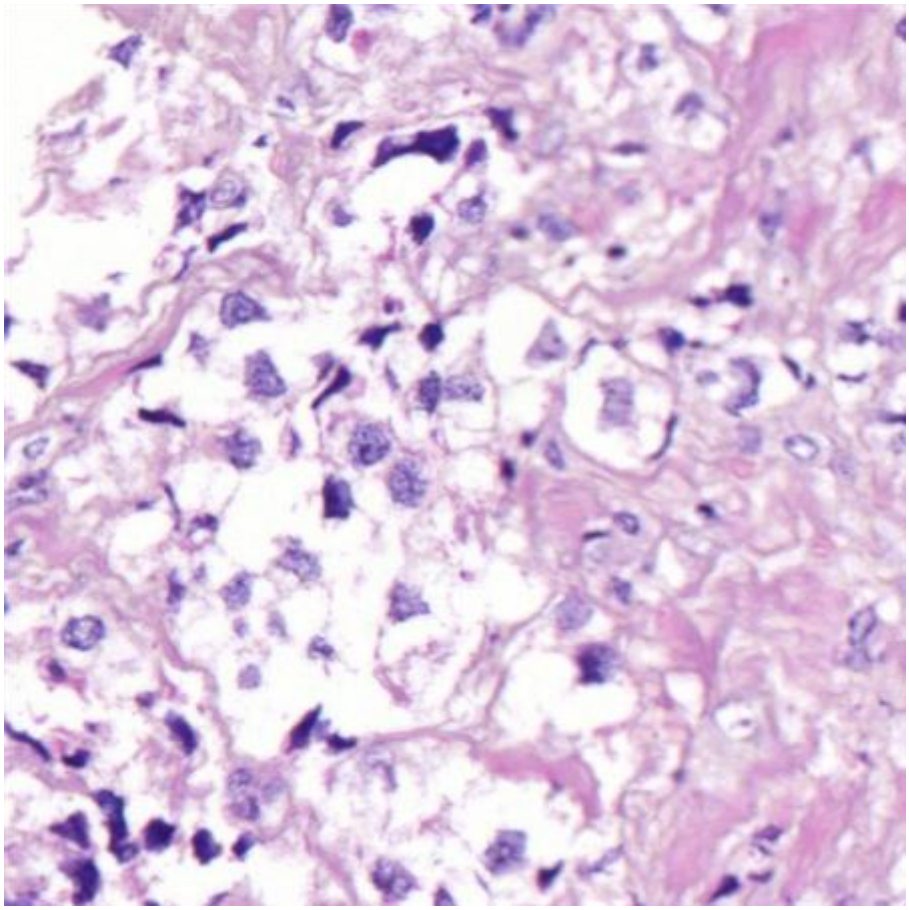
Wienert

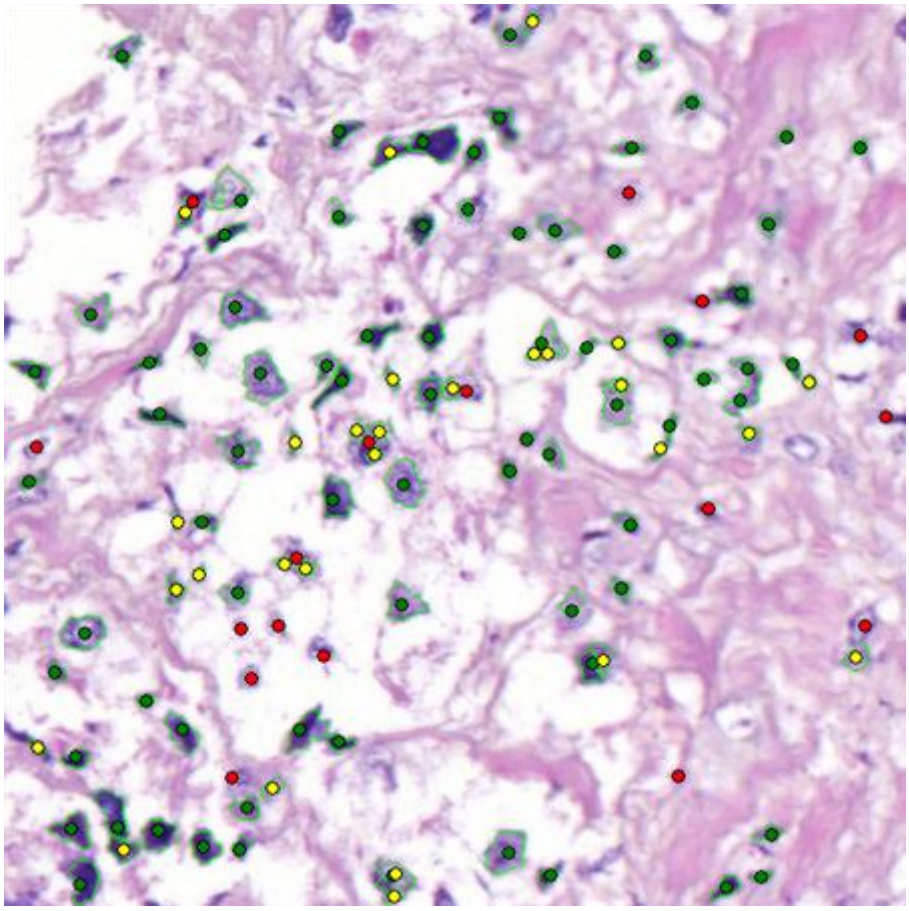


Al-Kofahi (optimized)

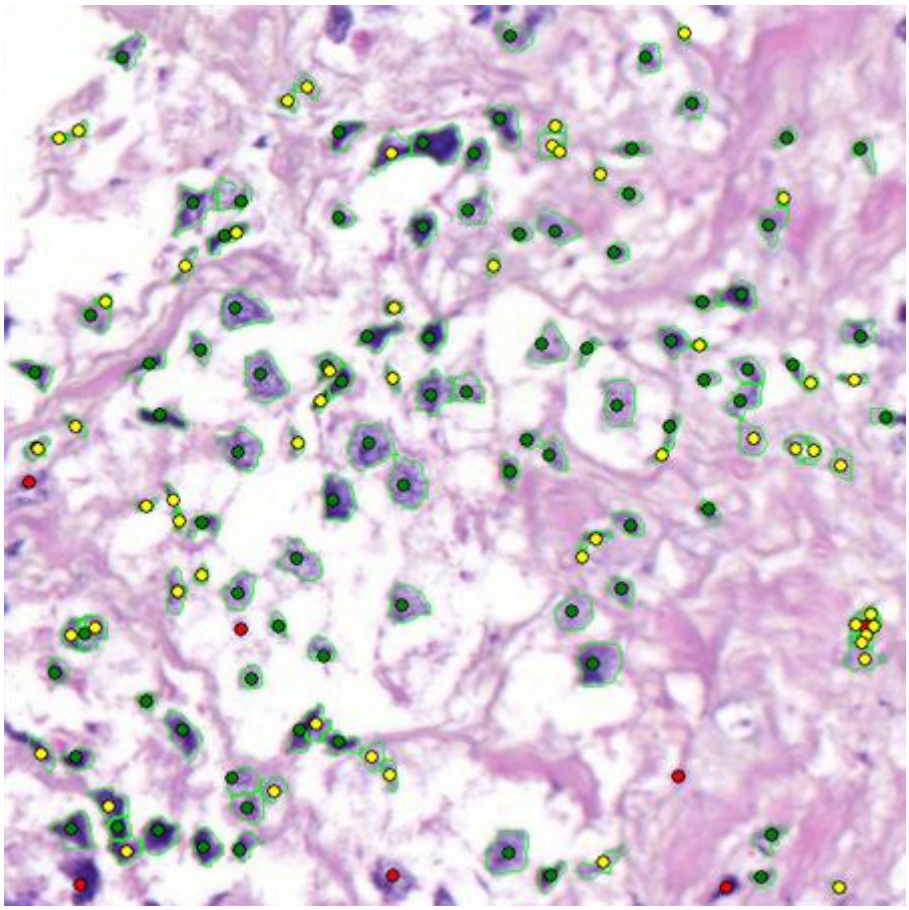
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	305	305	305
true positive count	240	265	265
false negative count	65	40	40
false positive count	14	49	22
precision	0,945	0,844	0,923
recall	0,787	0,869	0,869
conglomerate	0,991	0,989	0,985
time(ms)	639	780	2043

5





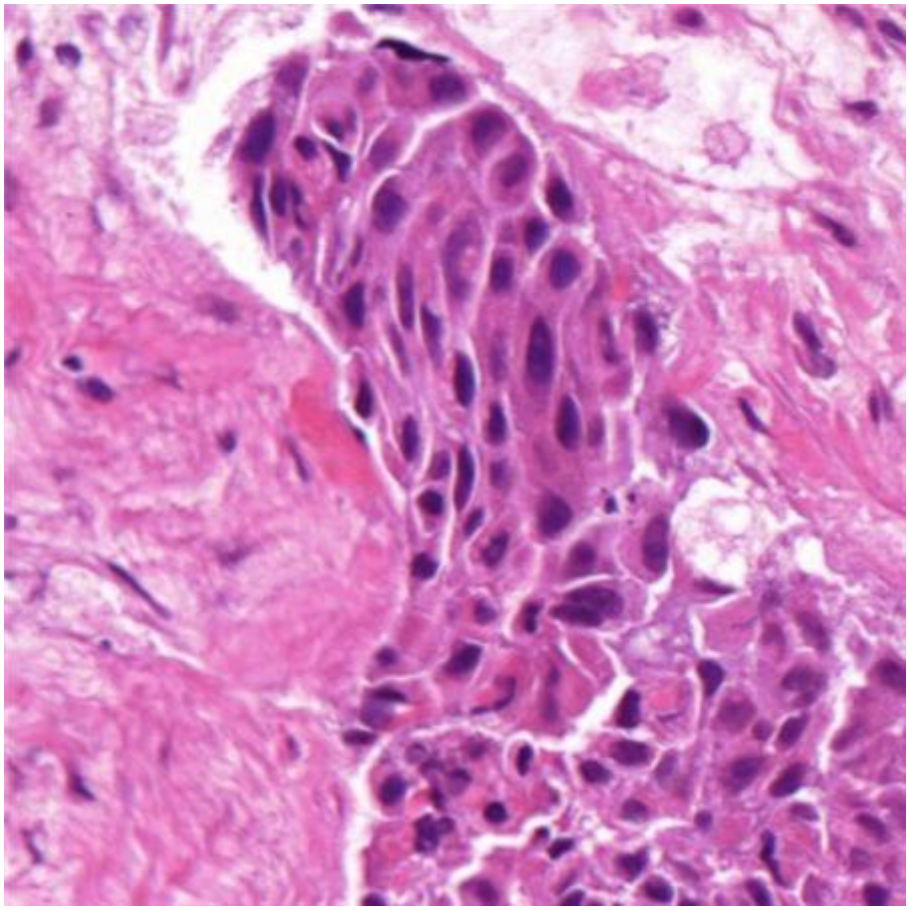
Wienert

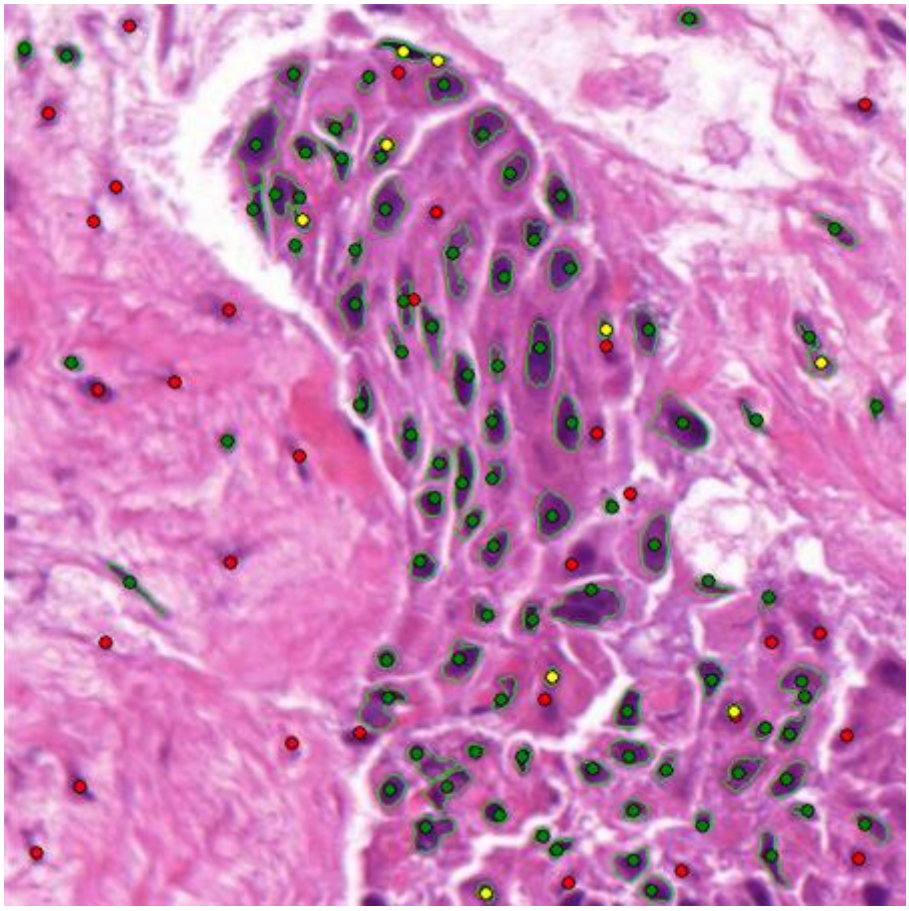


Al-Kofahi (optimized)

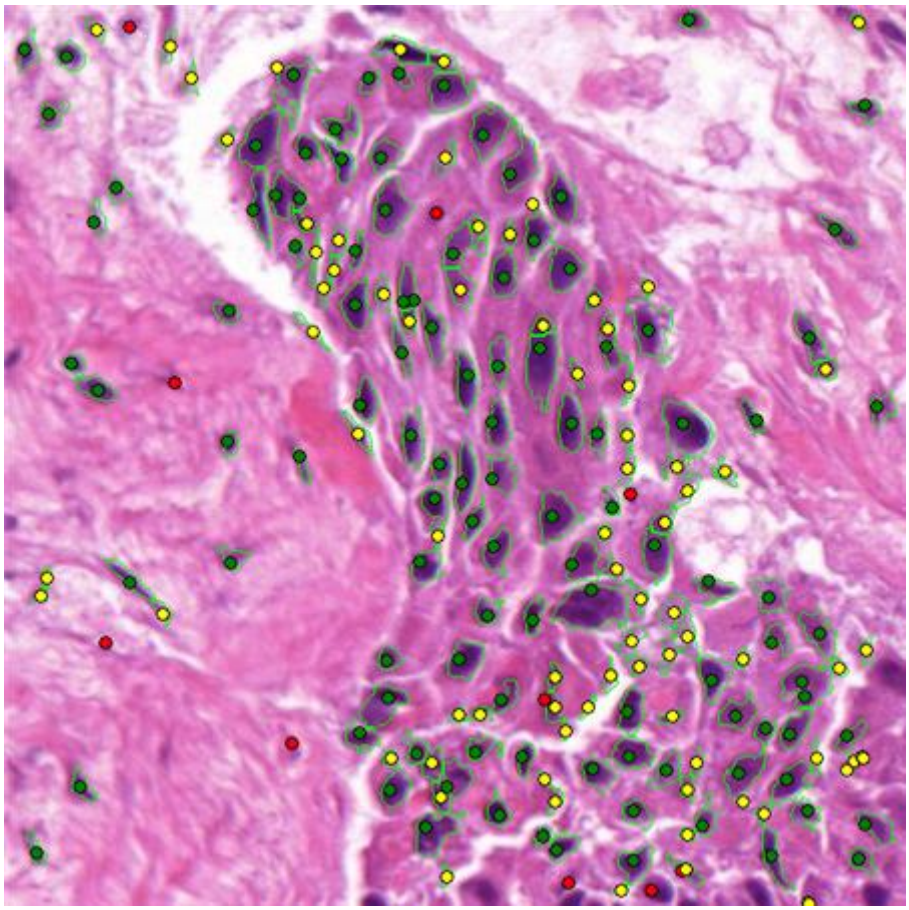
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	93	93	93
true positive count	76	85	85
false negative count	17	8	8
false positive count	28	48	54
precision	0,731	0,639	0,612
recall	0,817	0,914	0,914
conglomerate	0,987	0,988	1,000
time(ms)	608	780	1606

6





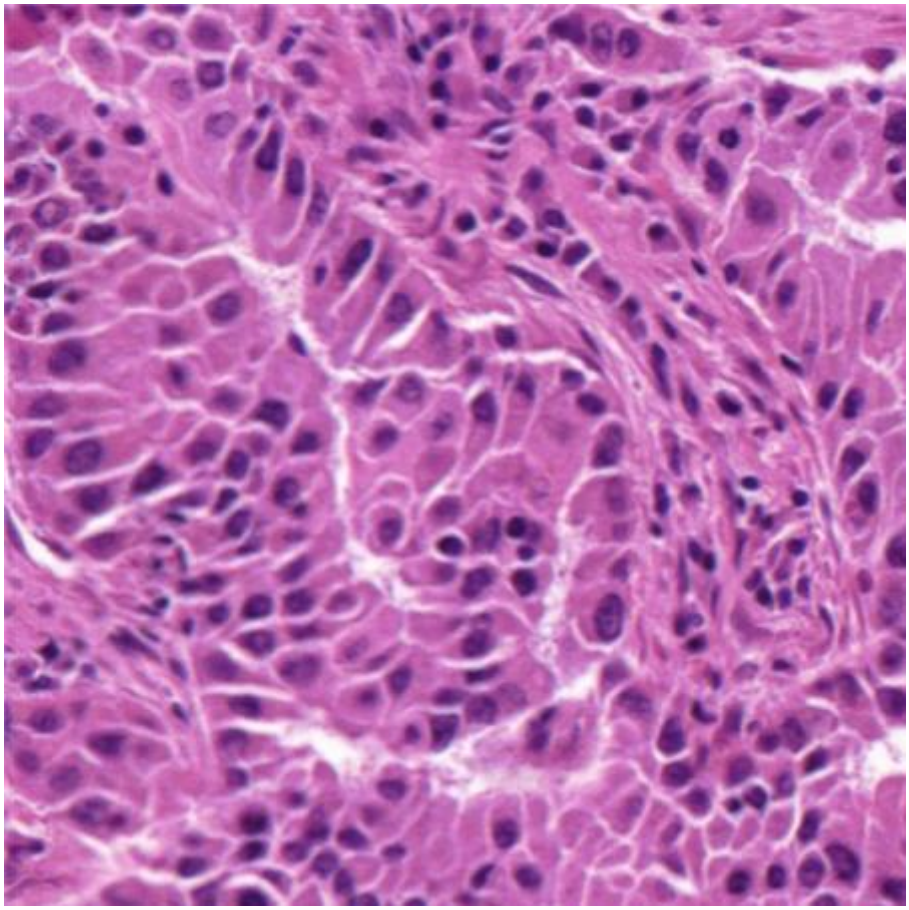
Wienert

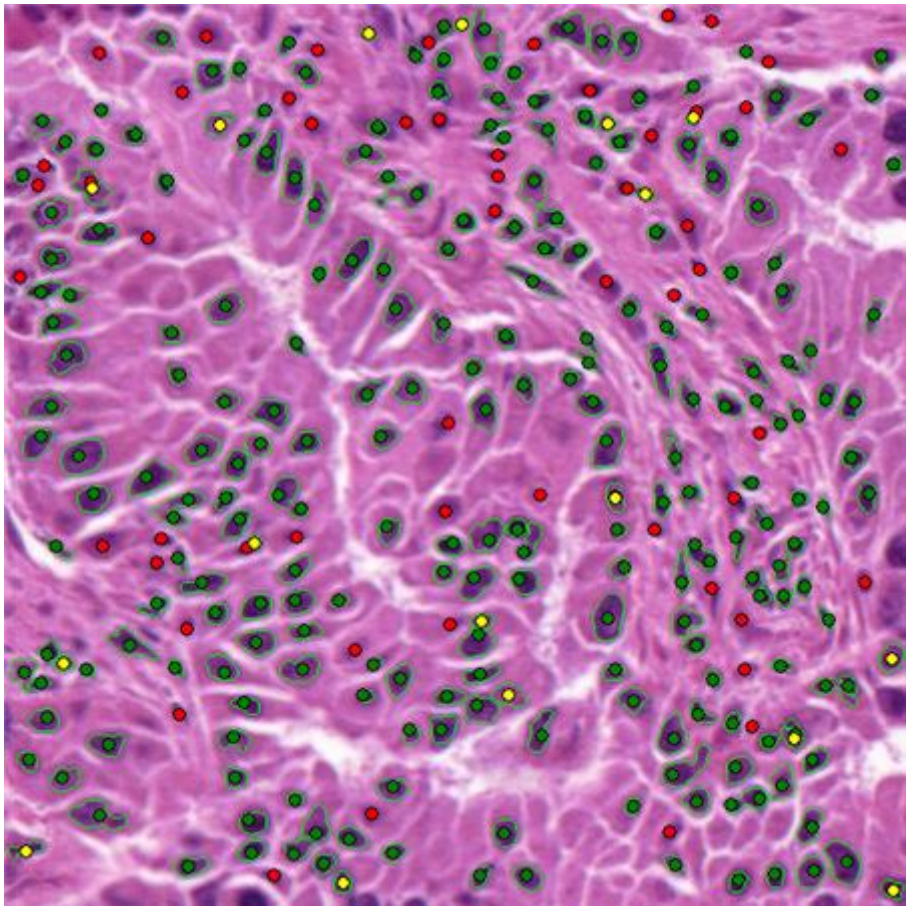


Al-Kofahi (optimized)

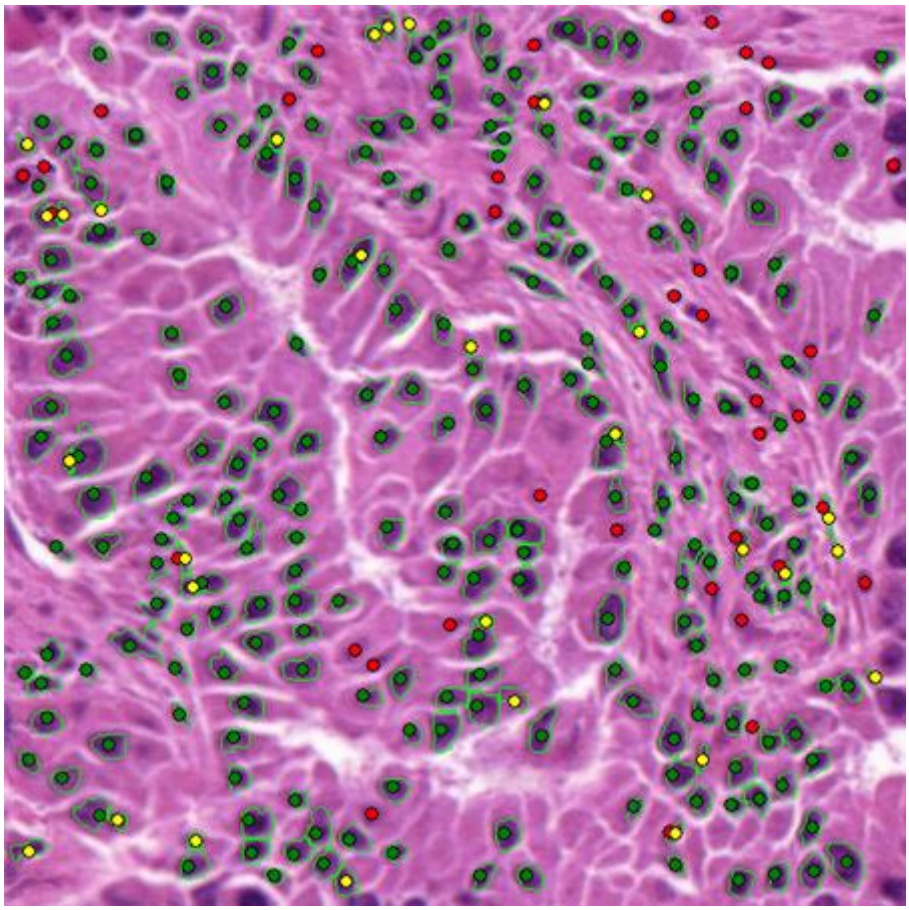
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	120	120	120
true positive count	90	111	110
false negative count	30	9	10
false positive count	9	141	78
precision	0,909	0,440	0,585
recall	0,750	0,925	0,917
conglomerate	0,978	0,973	0,973
time(ms)	608	624	1872

7





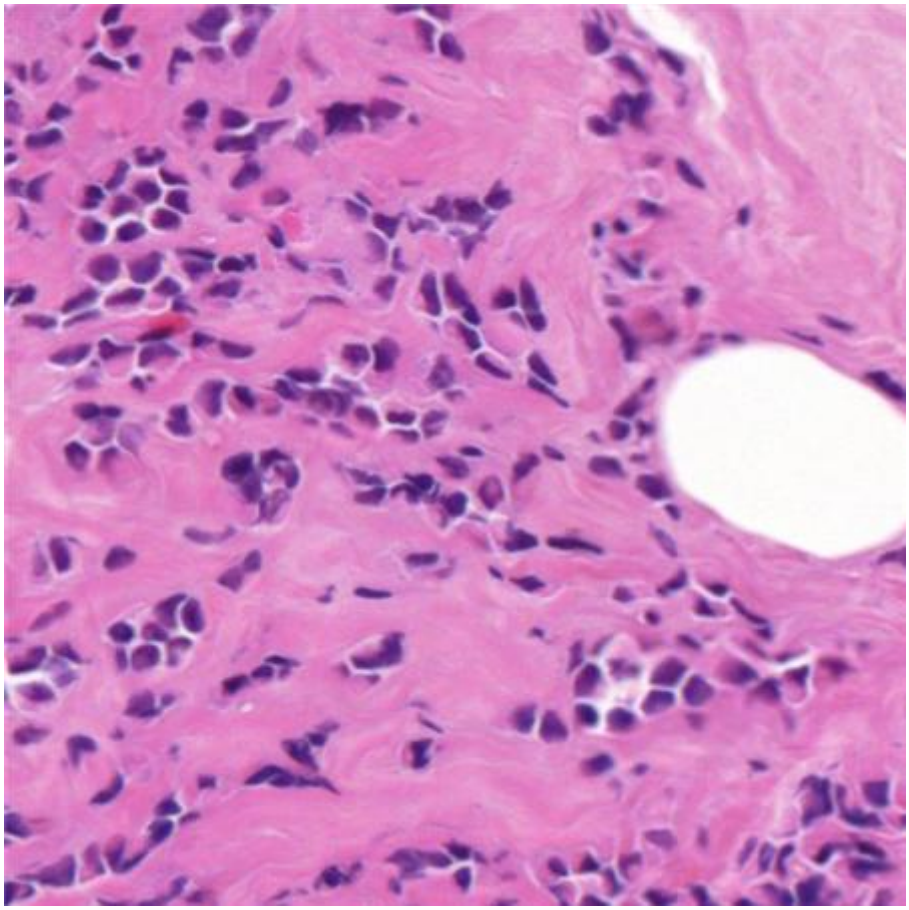
Wienert

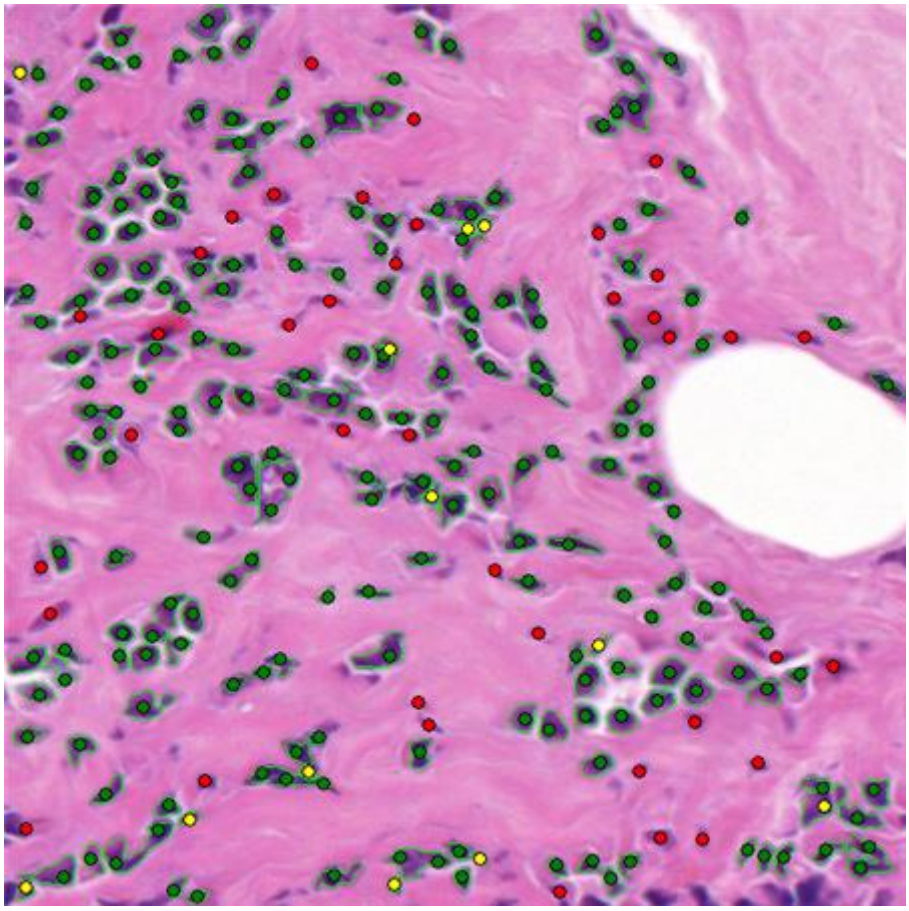


Al-Kofahi (optimized)

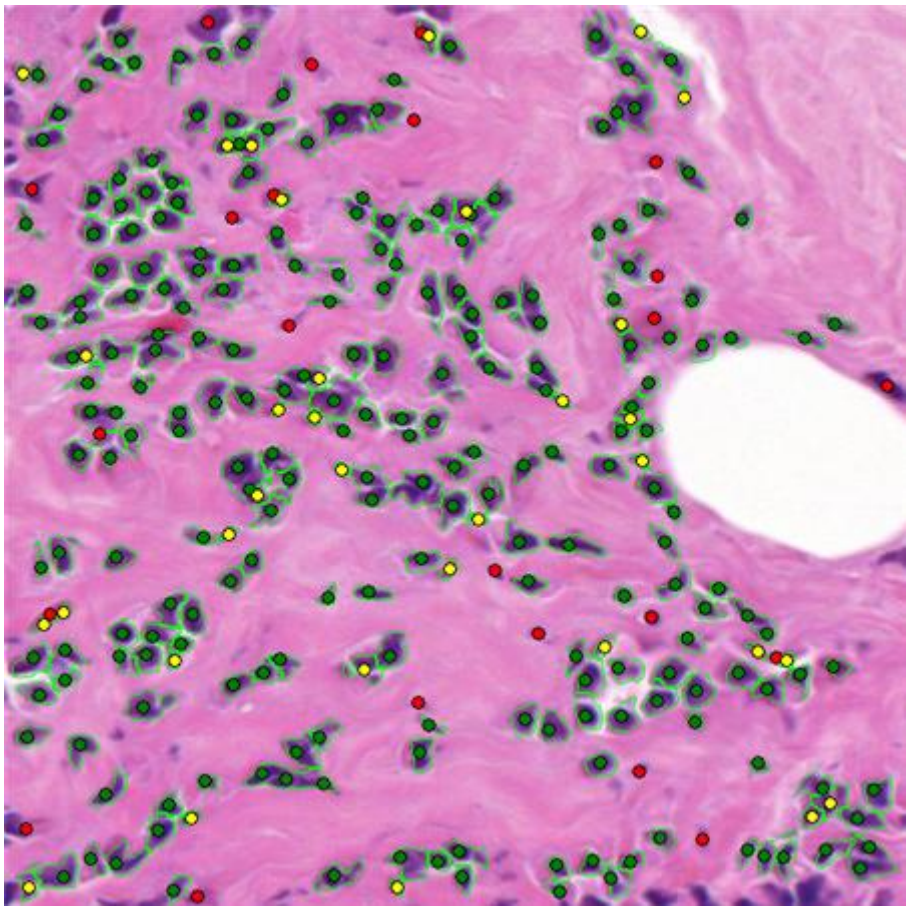
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	287	287	287
true positive count	231	249	249
false negative count	56	38	38
false positive count	17	54	30
precision	0,931	0,822	0,892
recall	0,805	0,868	0,868
conglomerate	0,983	0,984	0,984
time(ms)	639	748	2137

8





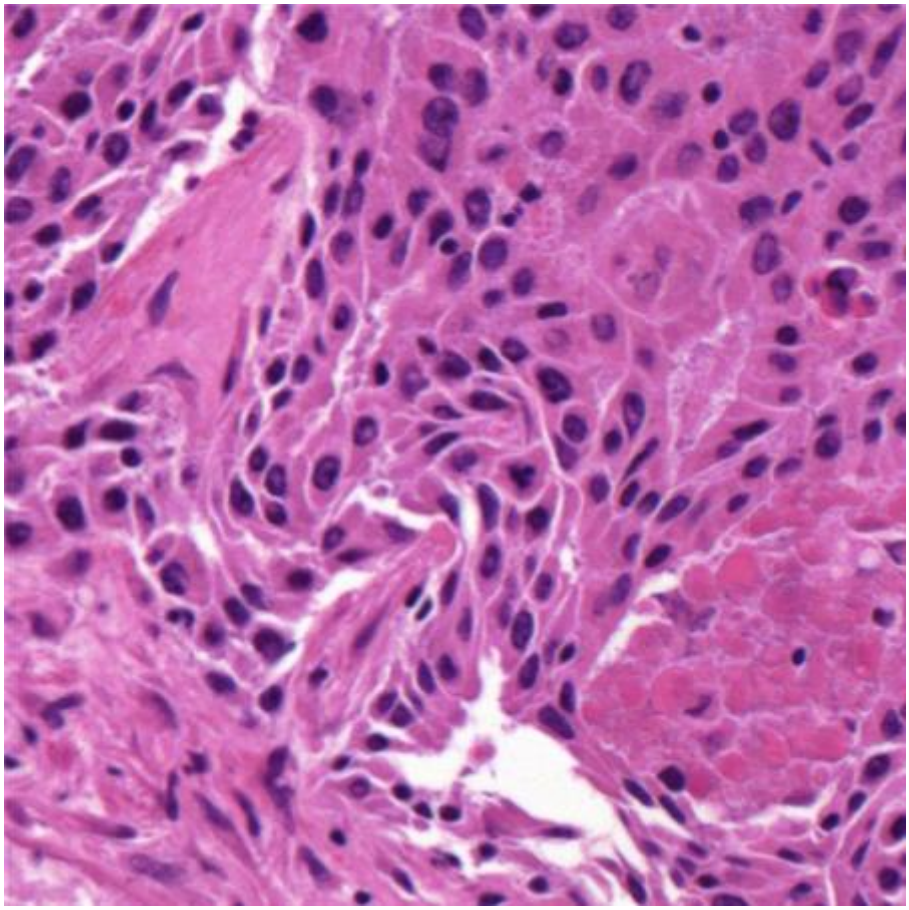
Wienert

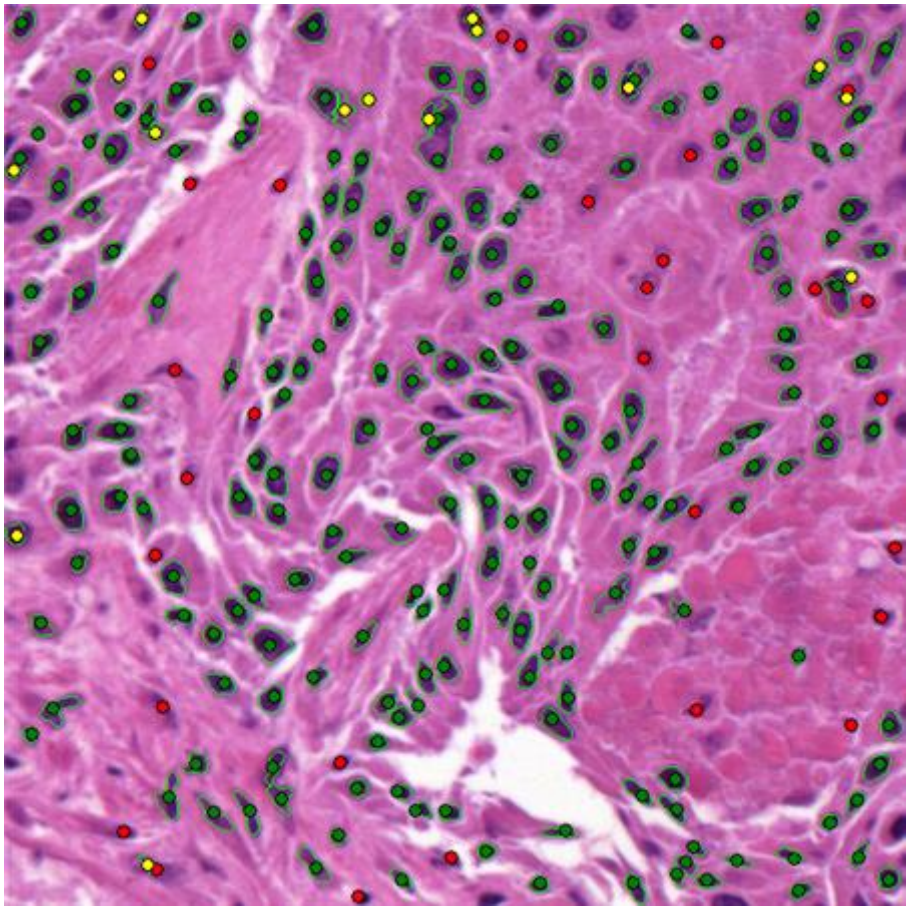


Al-Kofahi (optimized)

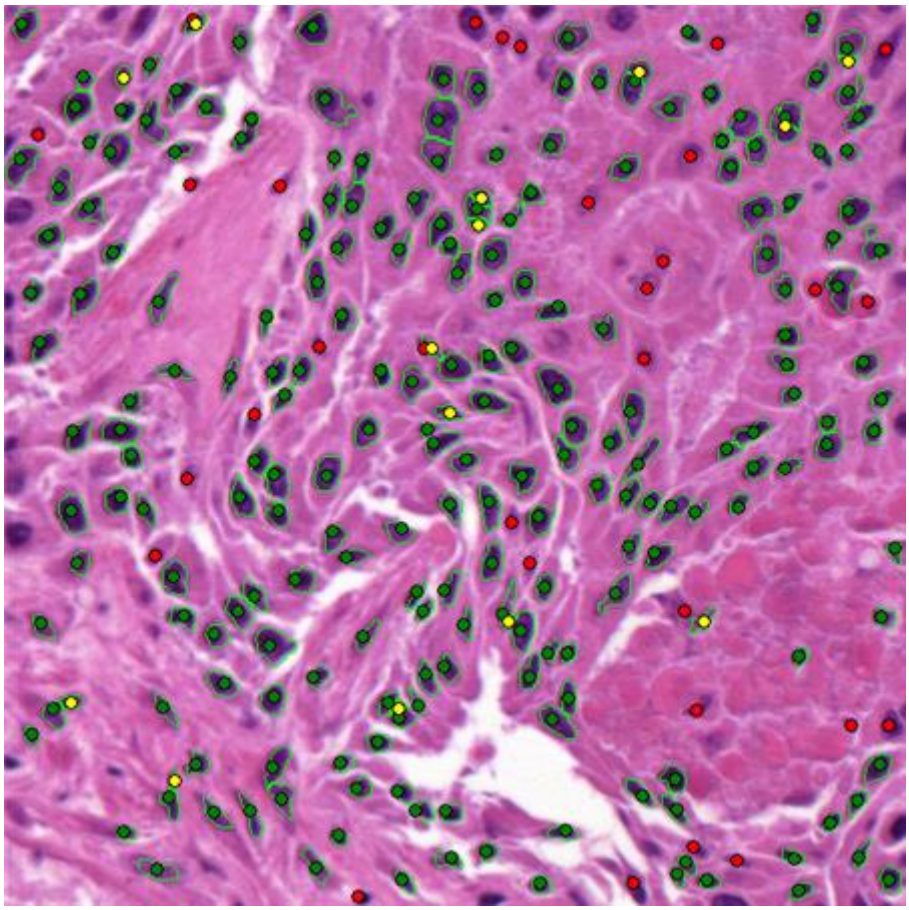
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	252	252	252
true positive count	213	228	229
false negative count	39	24	23
false positive count	12	84	33
precision	0,947	0,731	0,874
recall	0,845	0,905	0,909
conglomerate	0,906	0,917	0,913
time(ms)	717	655	1872

9





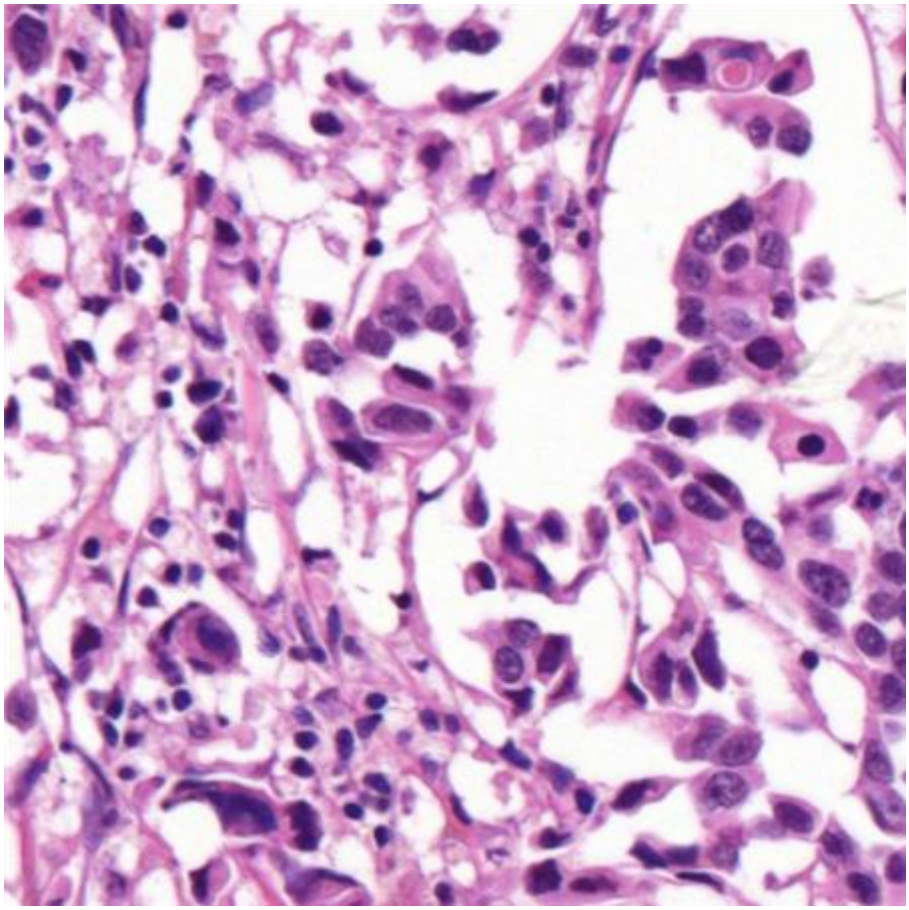
Wienert

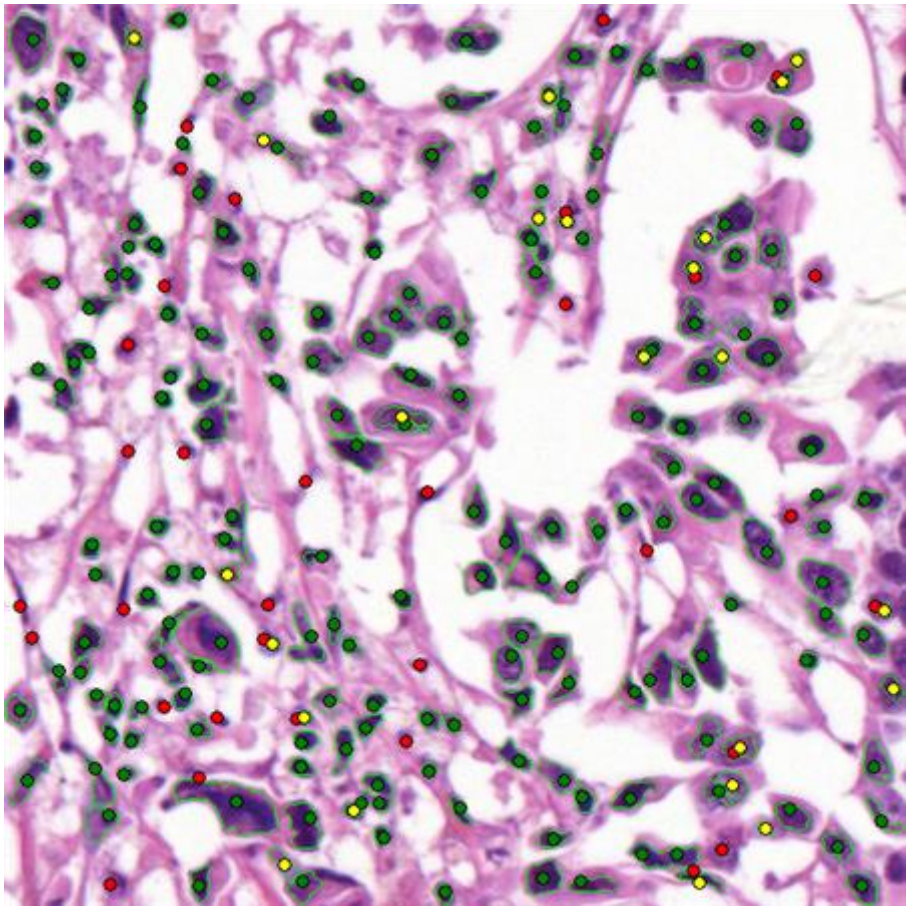


Al-Kofahi (optimized)

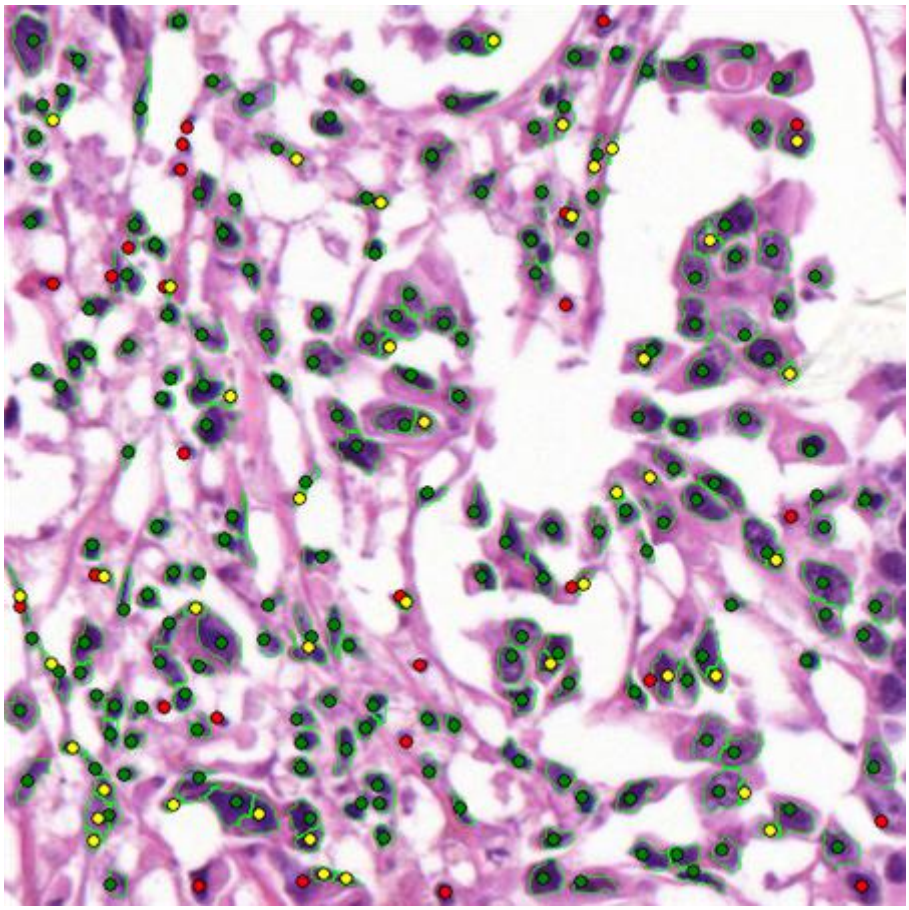
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	245	245	245
true positive count	214	213	214
false negative count	31	32	31
false positive count	15	37	14
precision	0,934	0,852	0,939
recall	0,873	0,869	0,873
conglomerate	0,986	0,972	0,981
time(ms)	686	608	1825

10





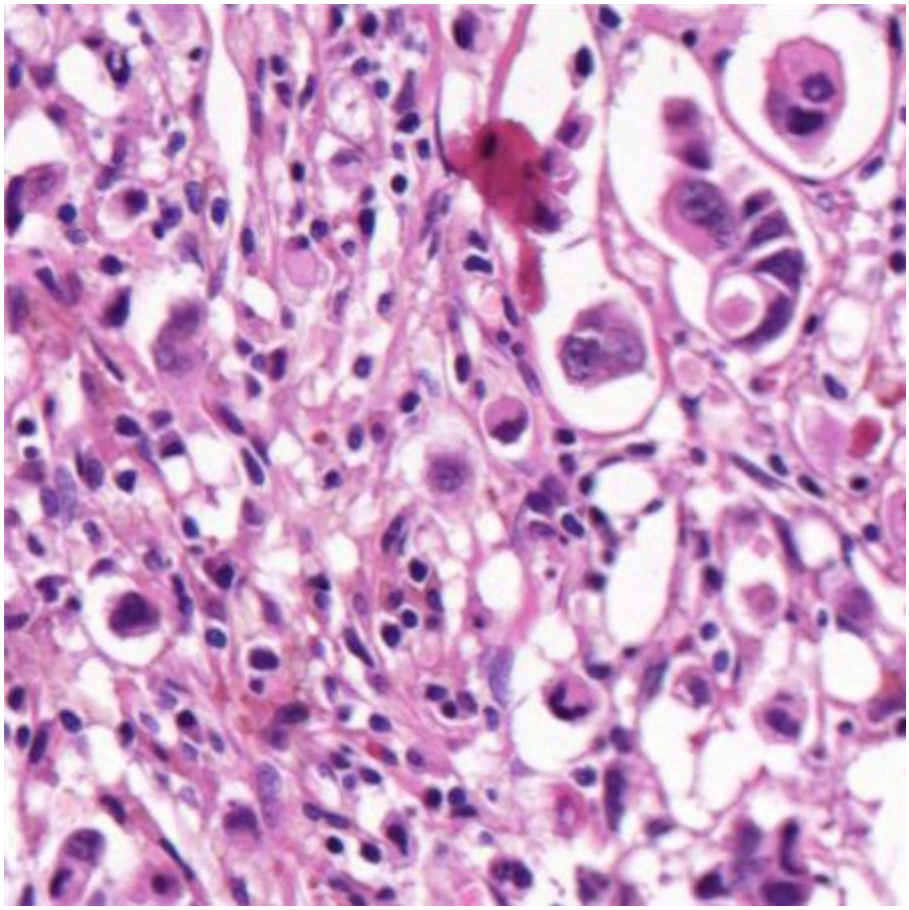
Wienert

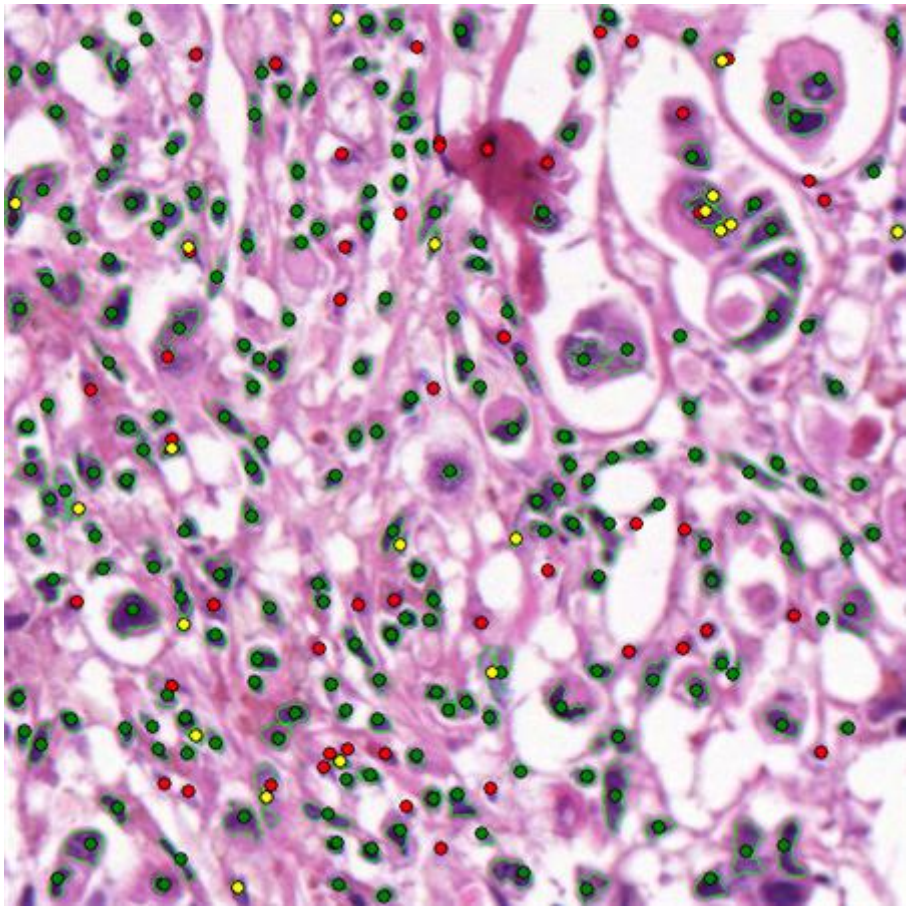


Al-Kofahi (optimized)

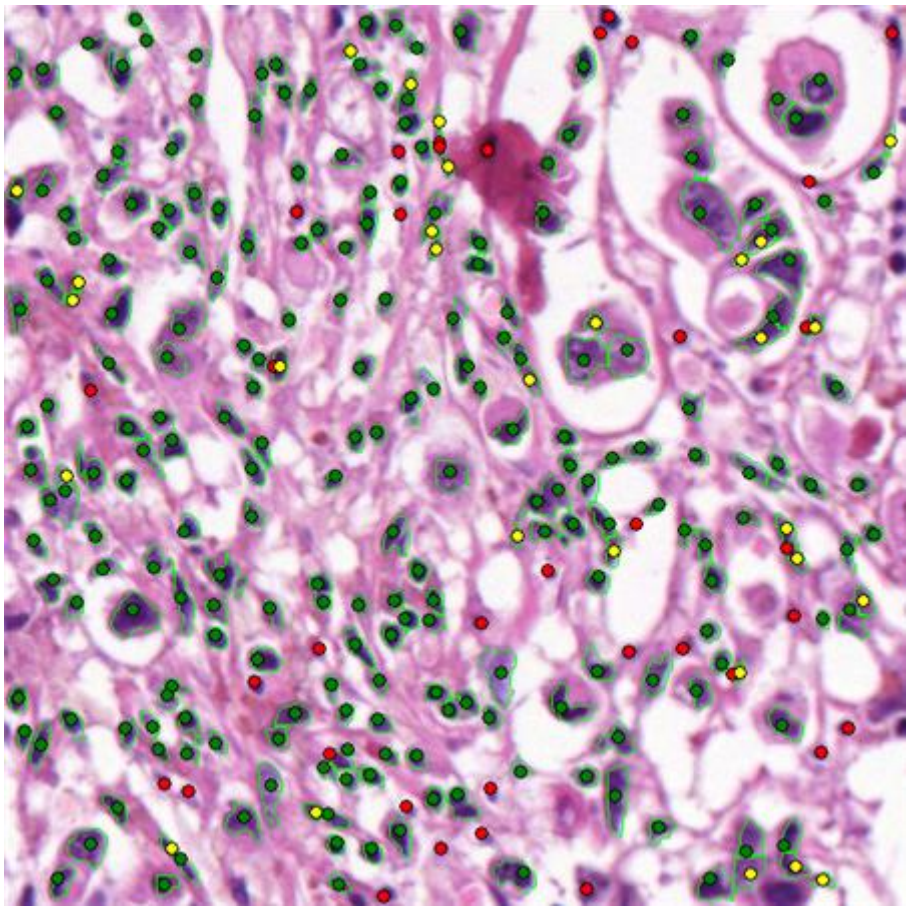
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	213	213	213
true positive count	180	186	186
false negative count	33	27	27
false positive count	23	59	41
precision	0,887	0,759	0,819
recall	0,845	0,873	0,873
conglomerate	0,956	0,989	0,989
time(ms)	764	702	1825

11





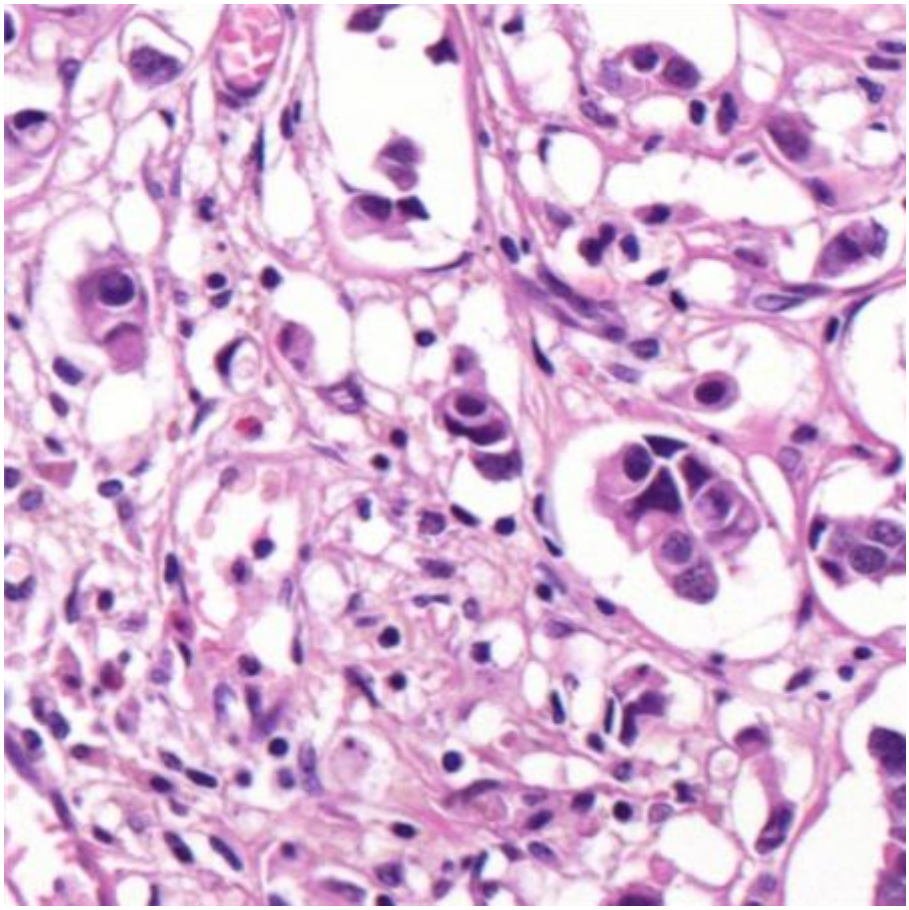
Wienert

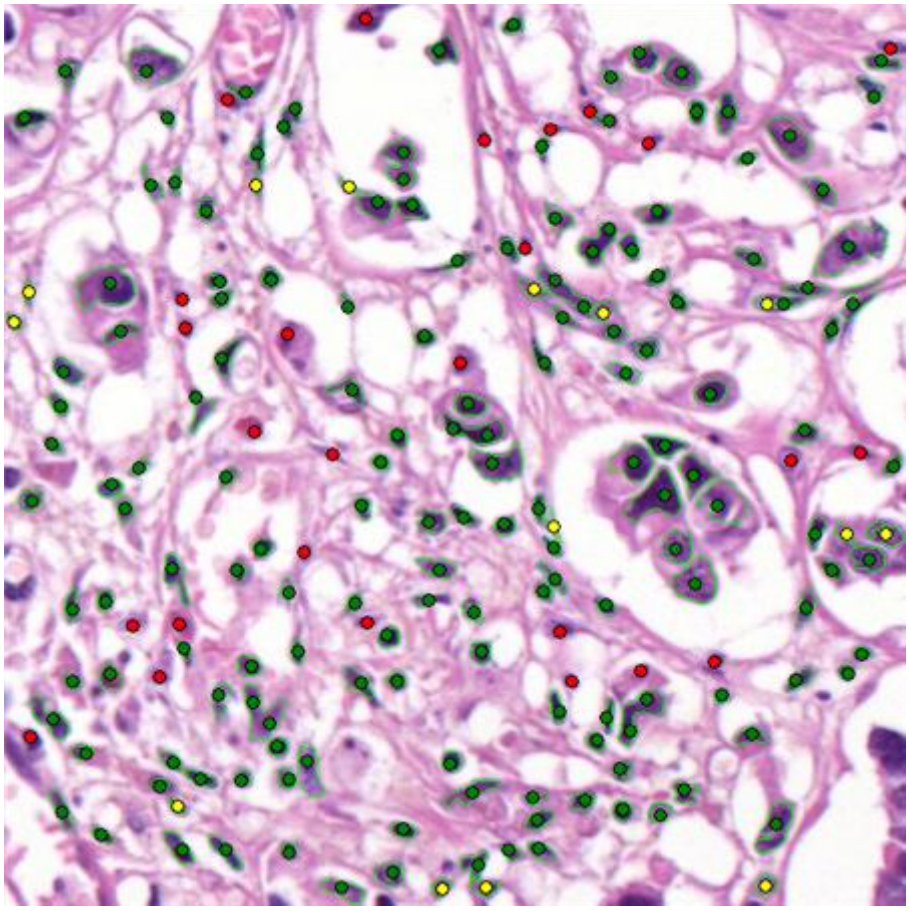


Al-Kofahi (optimized)

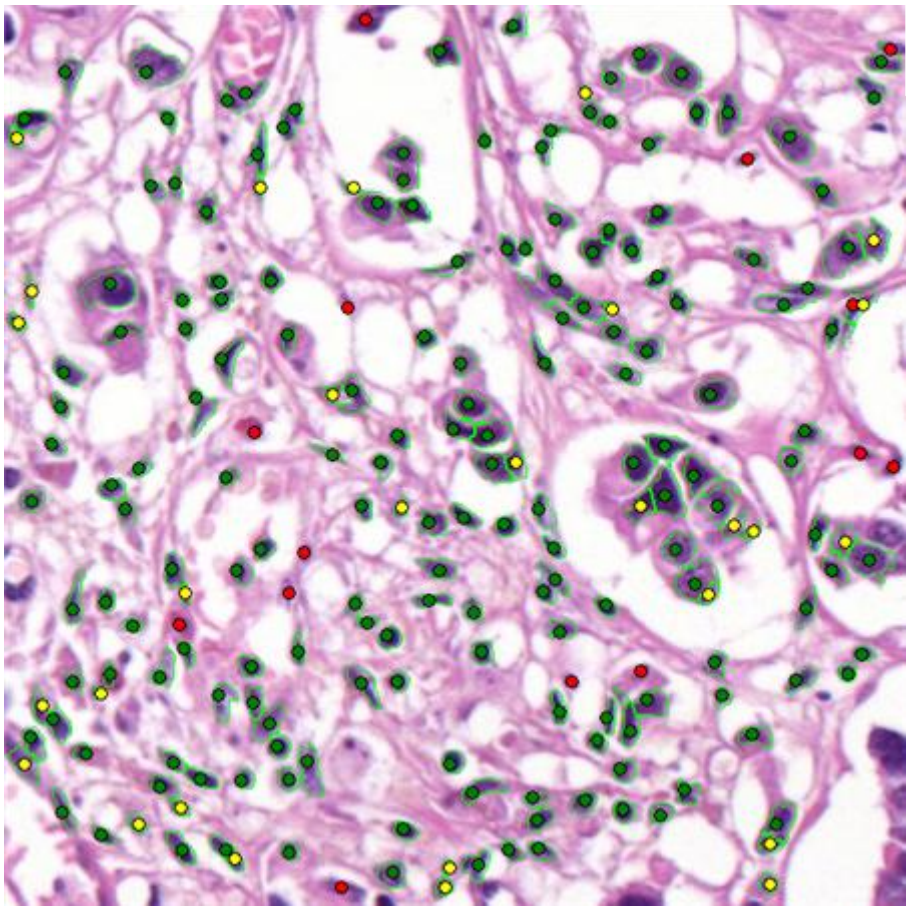
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	257	257	257
true positive count	208	224	221
false negative count	49	33	36
false positive count	20	56	29
precision	0,912	0,800	0,884
recall	0,809	0,872	0,860
conglomerate	0,976	0,969	0,973
time(ms)	733	639	1950

12





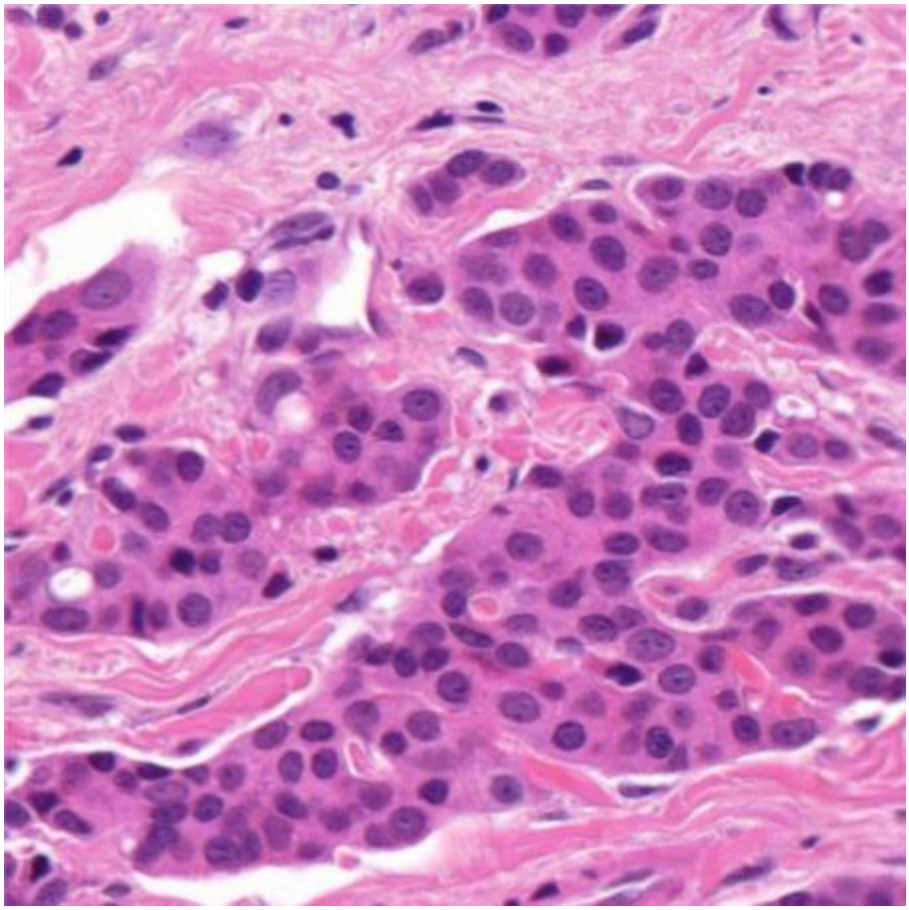
Wienert

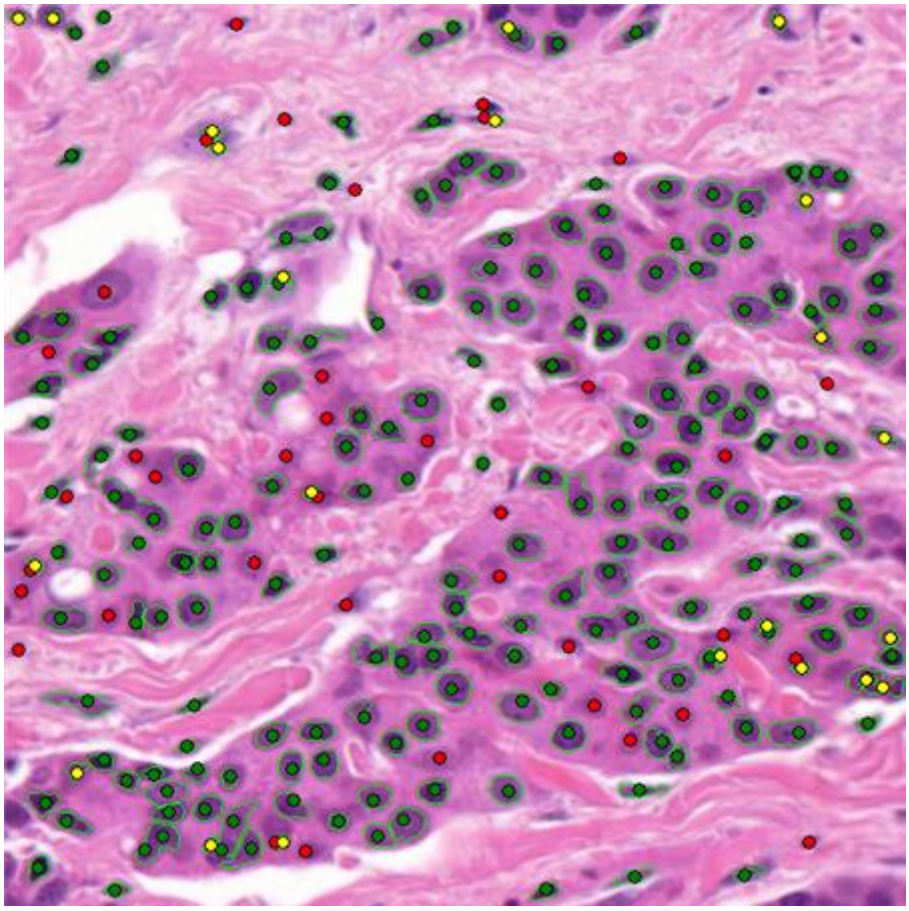


Al-Kofahi (optimized)

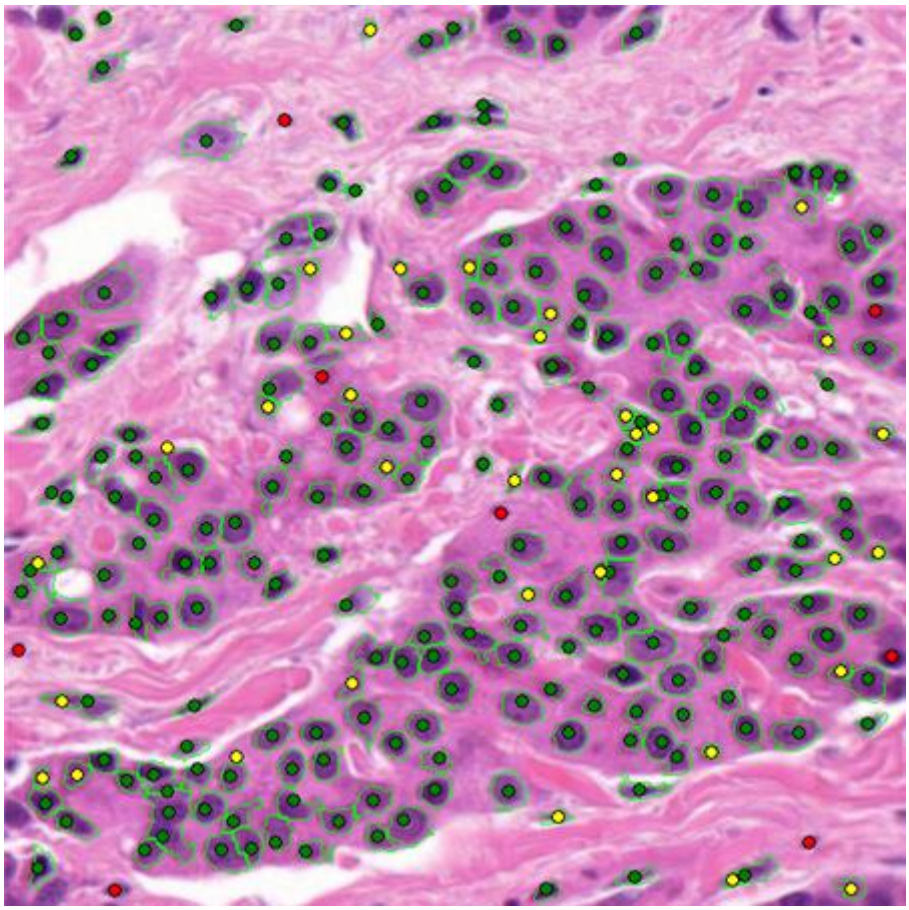
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	186	186	186
true positive count	160	170	172
false negative count	26	16	14
false positive count	14	51	28
precision	0,920	0,769	0,860
recall	0,860	0,914	0,925
conglomerate	0,950	0,947	0,953
time(ms)	608	764	1762

13





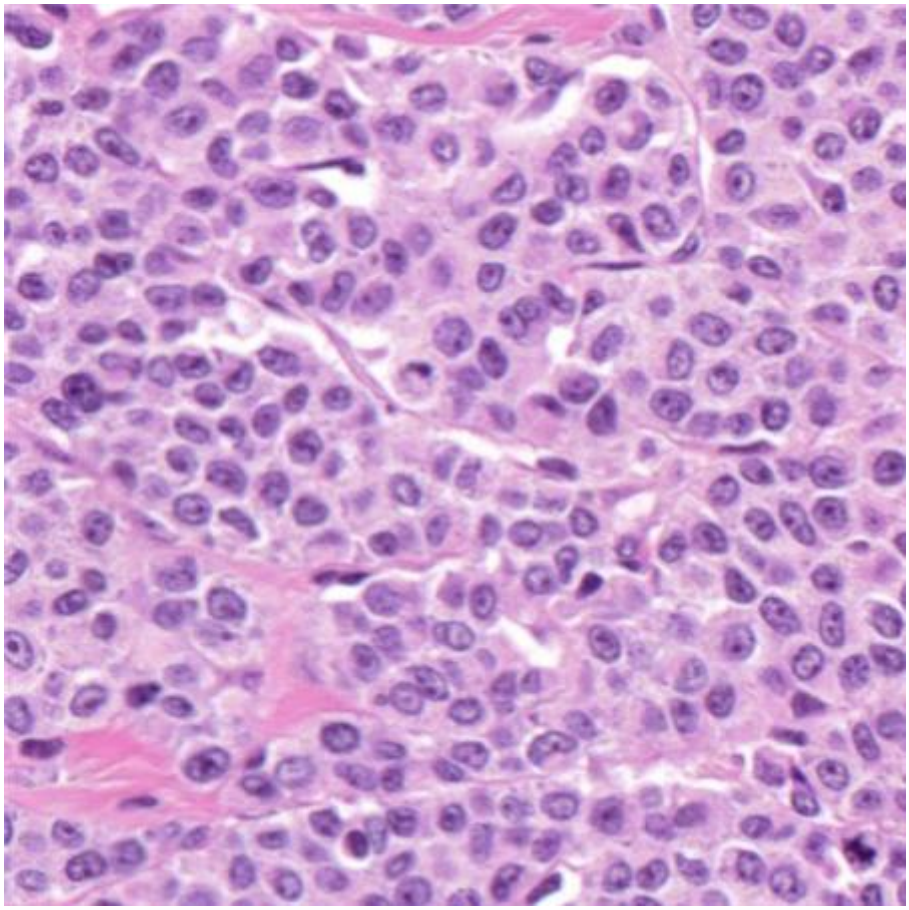
Wienert

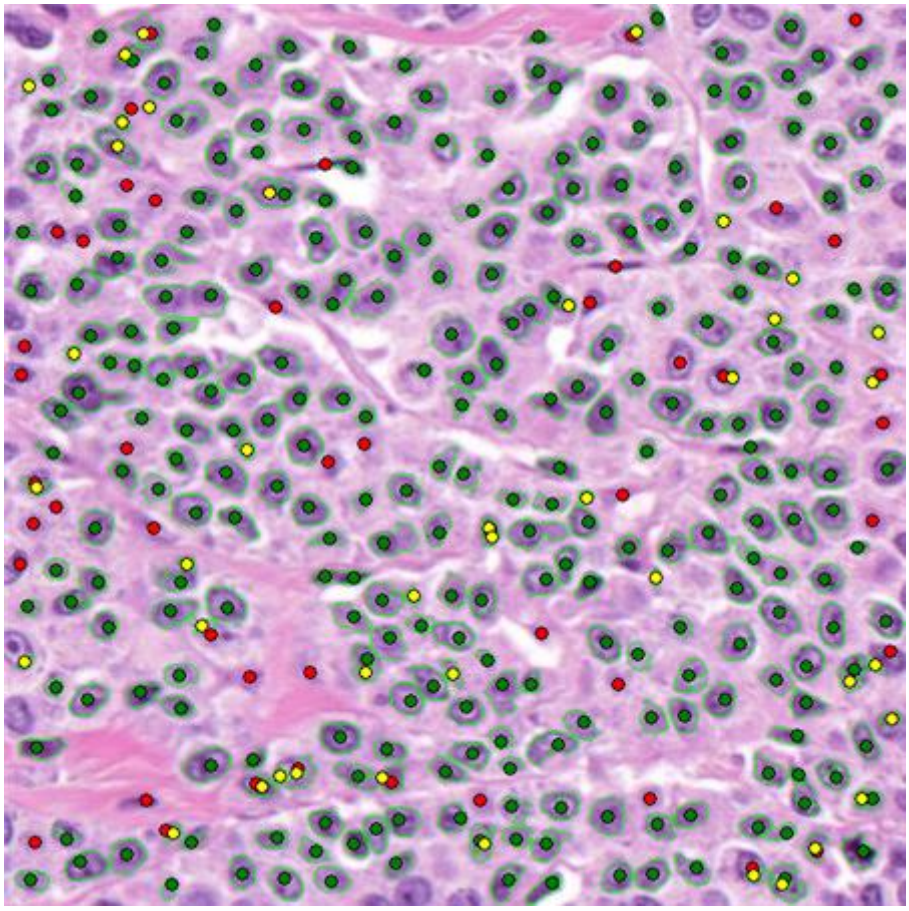


Al-Kofahi (optimized)

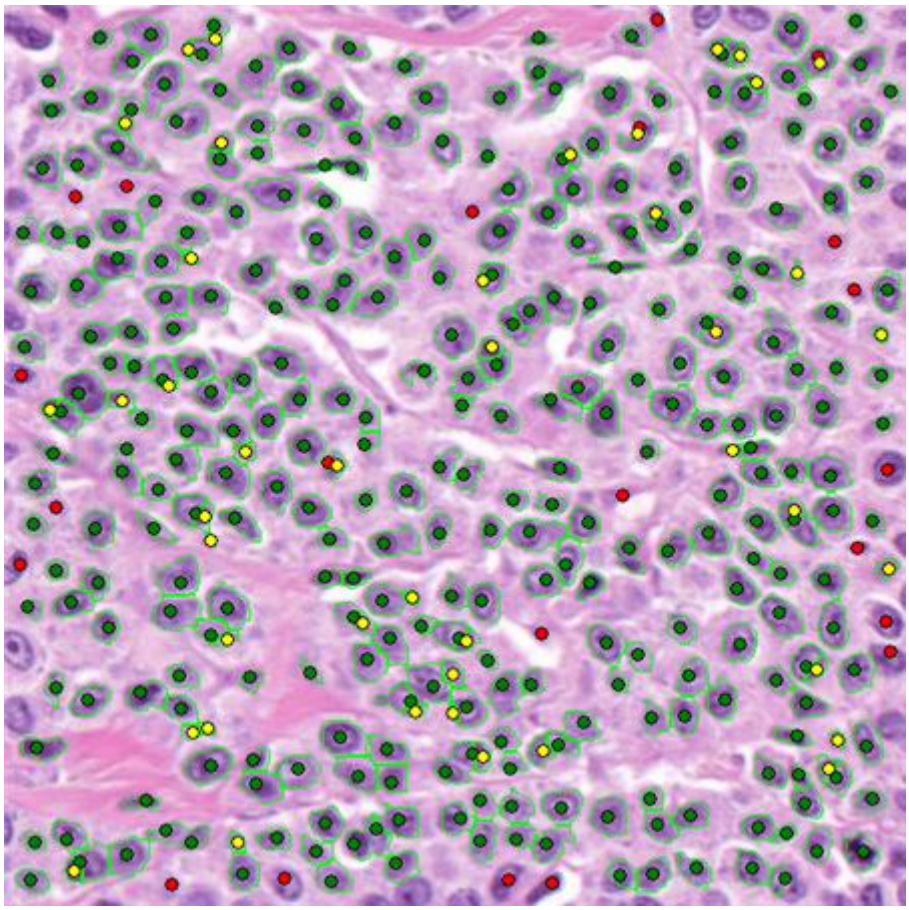
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	236	236	236
true positive count	197	228	228
false negative count	39	8	8
false positive count	22	94	35
precision	0,900	0,708	0,867
recall	0,835	0,966	0,966
conglomerate	0,939	0,956	0,961
time(ms)	748	639	1981

14





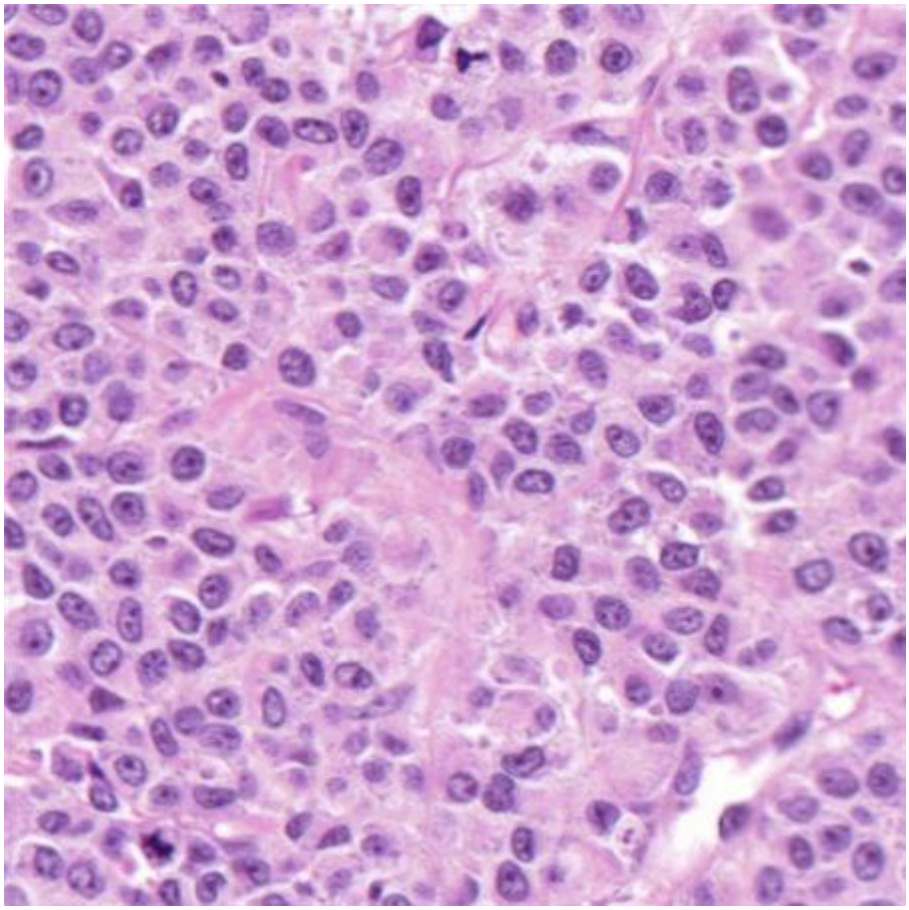
Wienert

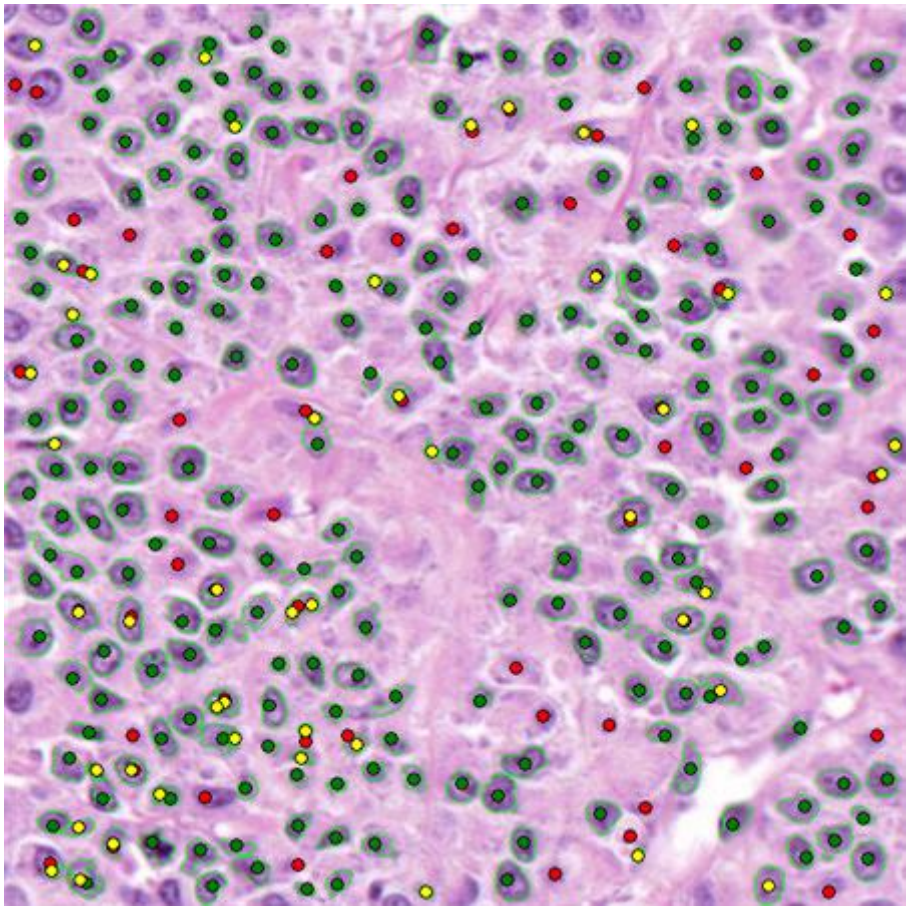


Al-Kofahi (optimized)

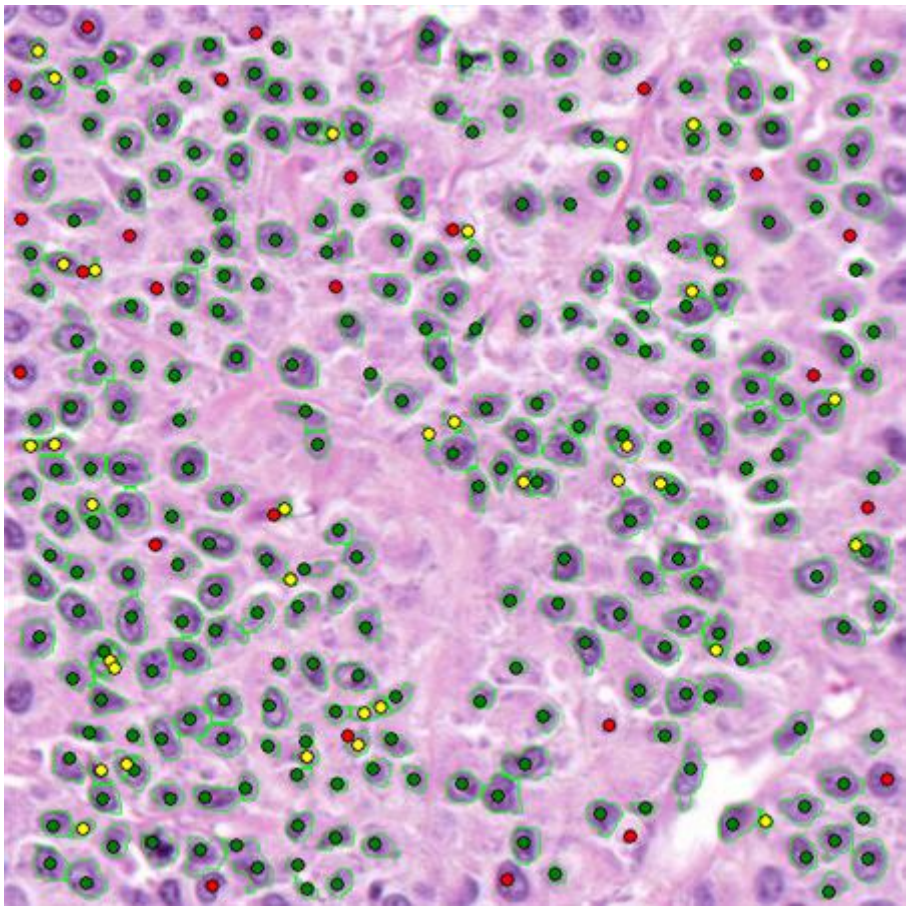
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	329	329	329
true positive count	281	308	307
false negative count	48	21	22
false positive count	41	54	43
precision	0,873	0,851	0,877
recall	0,854	0,936	0,933
conglomerate	0,968	0,981	0,984
time(ms)	1092	717	2371

15





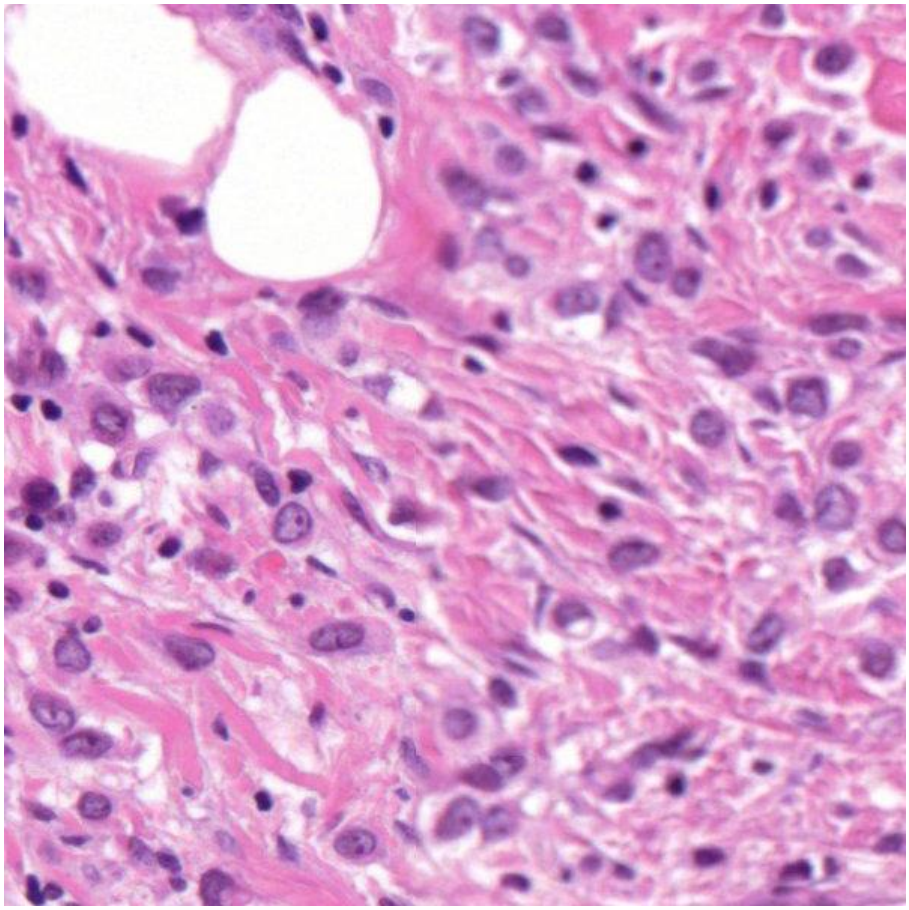
Wienert

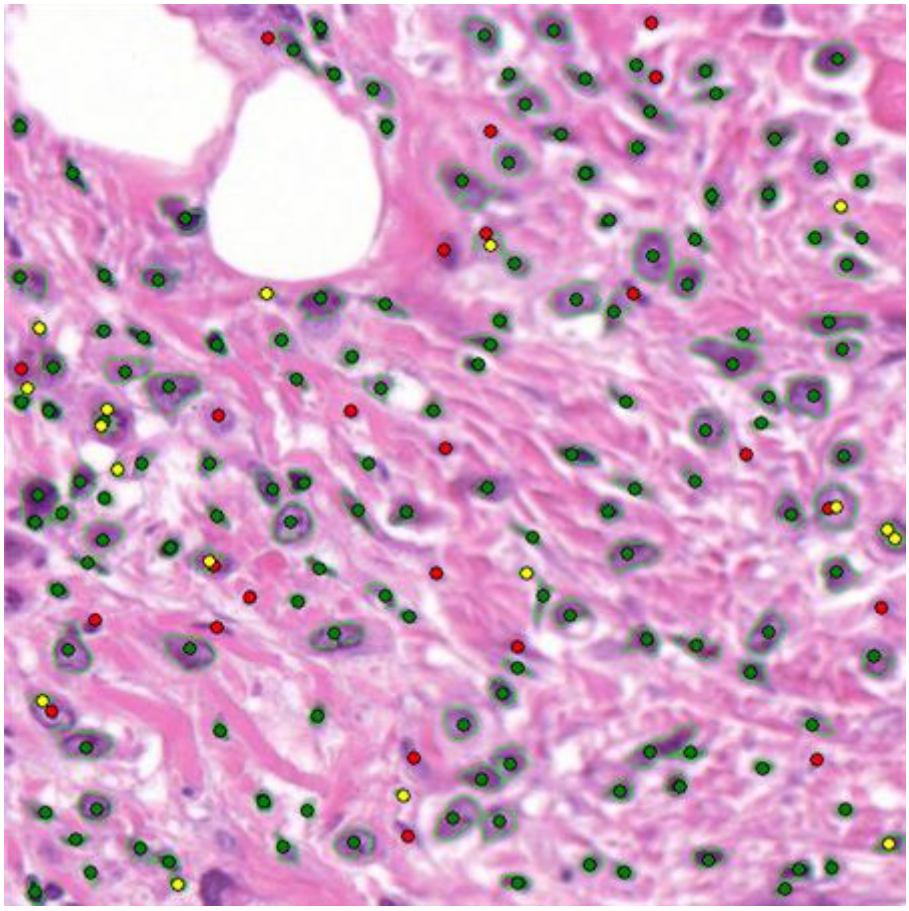


Al-Kofahi (optimized)

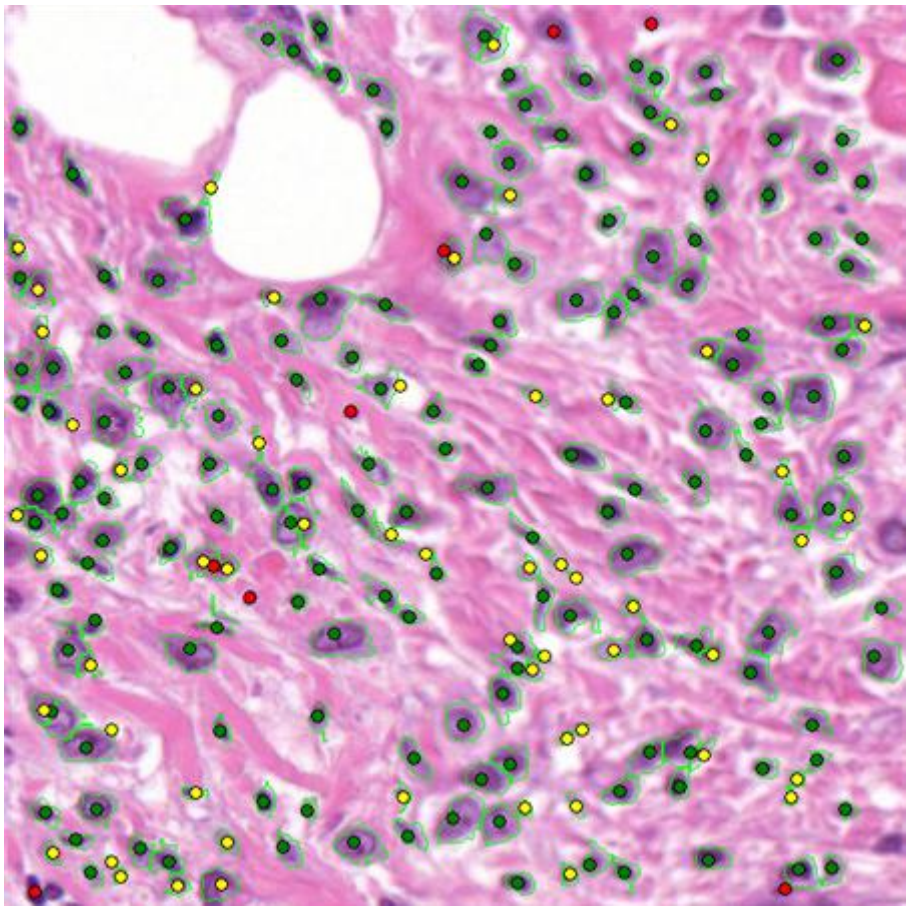
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	284	284	284
true positive count	228	259	259
false negative count	56	25	25
false positive count	47	49	35
precision	0,829	0,841	0,881
recall	0,803	0,912	0,912
conglomerate	0,969	0,988	0,988
time(ms)	920	733	2184

16





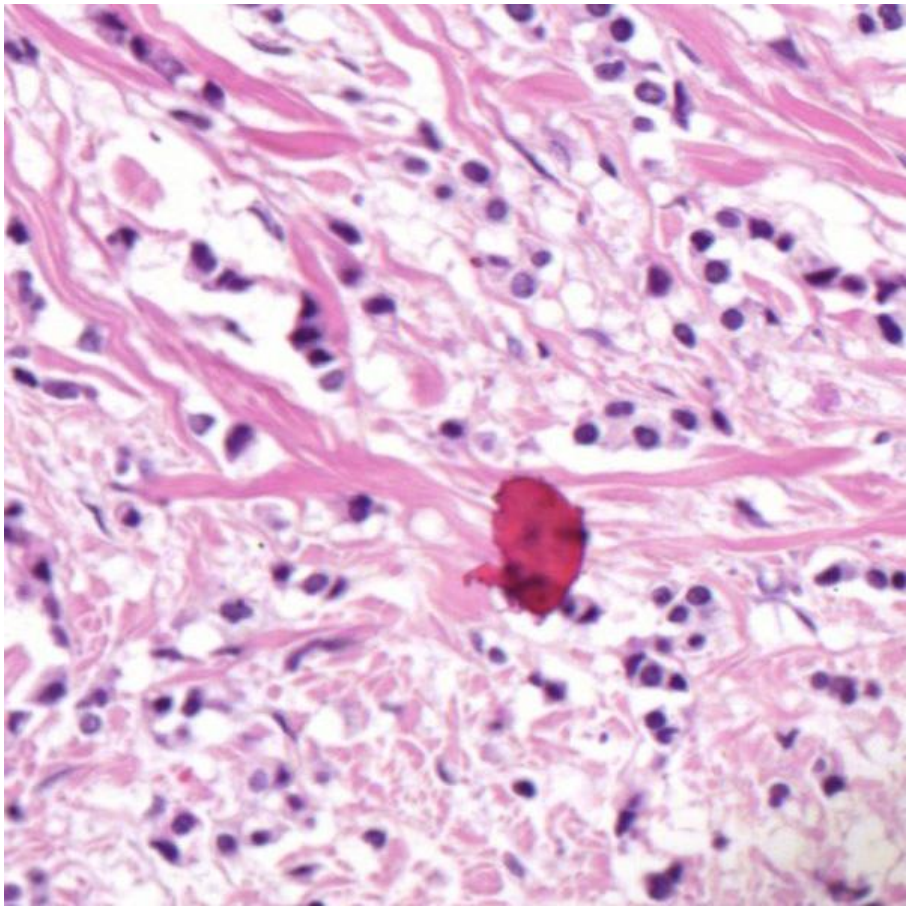
Wienert

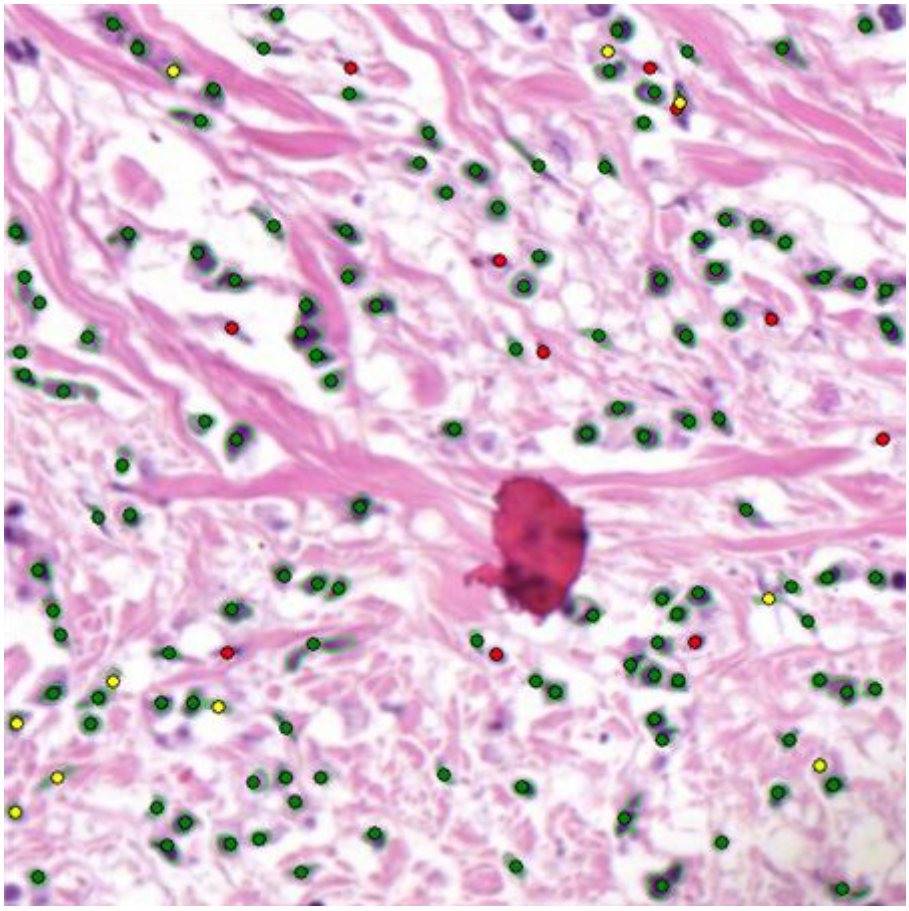


Al-Kofahi (optimized)

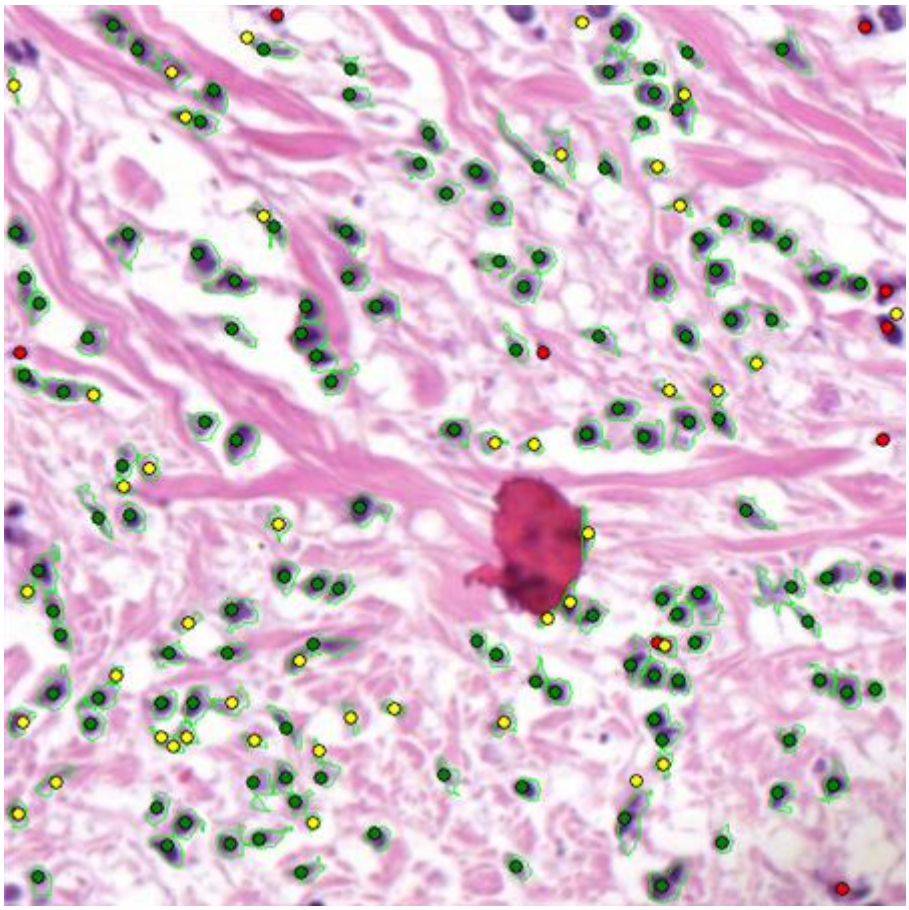
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	171	171	171
true positive count	147	161	163
false negative count	24	10	8
false positive count	17	103	57
precision	0,896	0,610	0,741
recall	0,860	0,942	0,953
conglomerate	0,993	0,994	0,988
time(ms)	686	655	1872

17





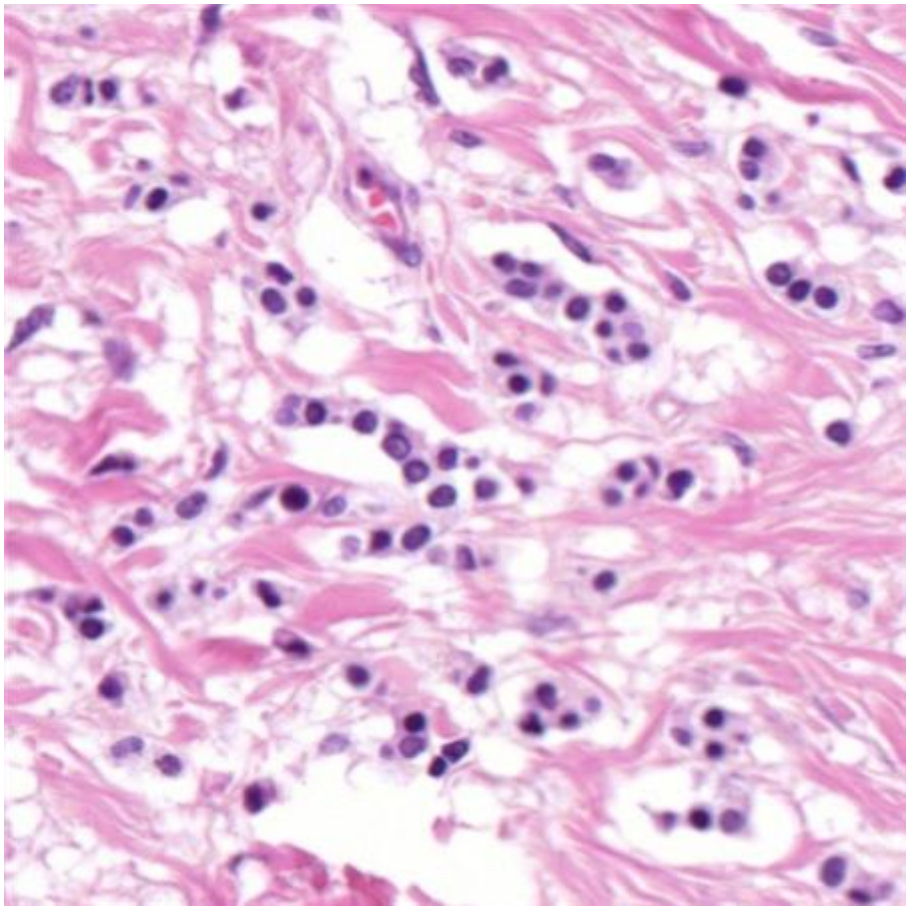
Wienert

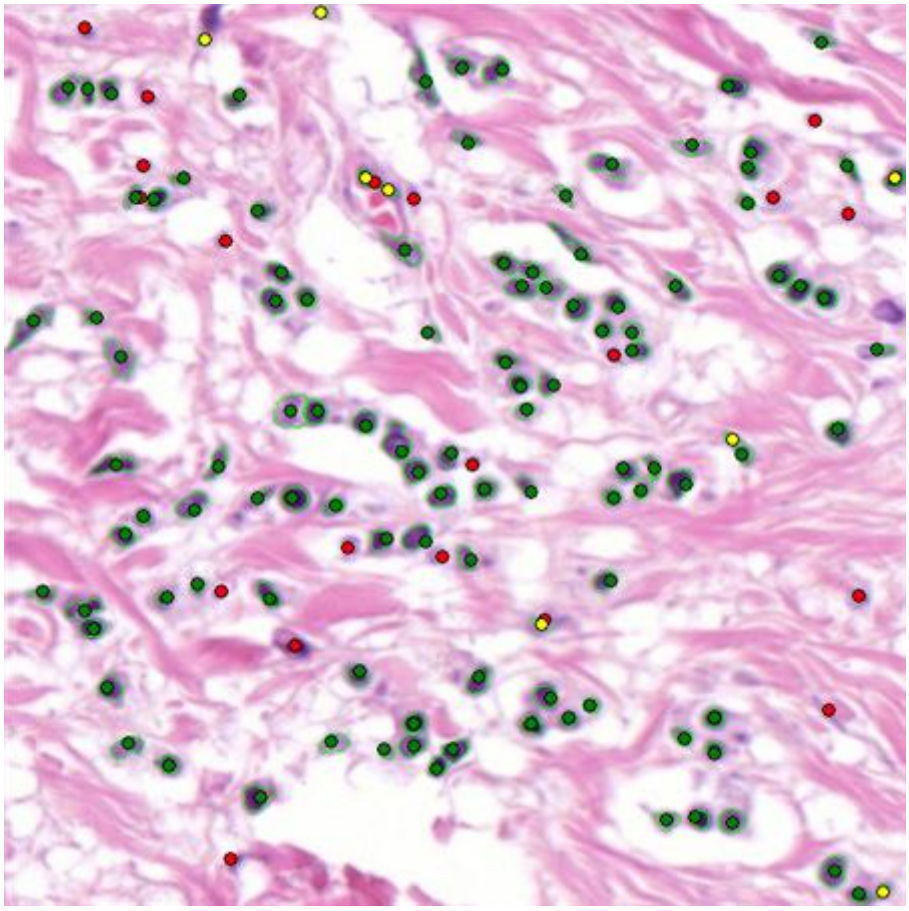


Al-Kofahi (optimized)

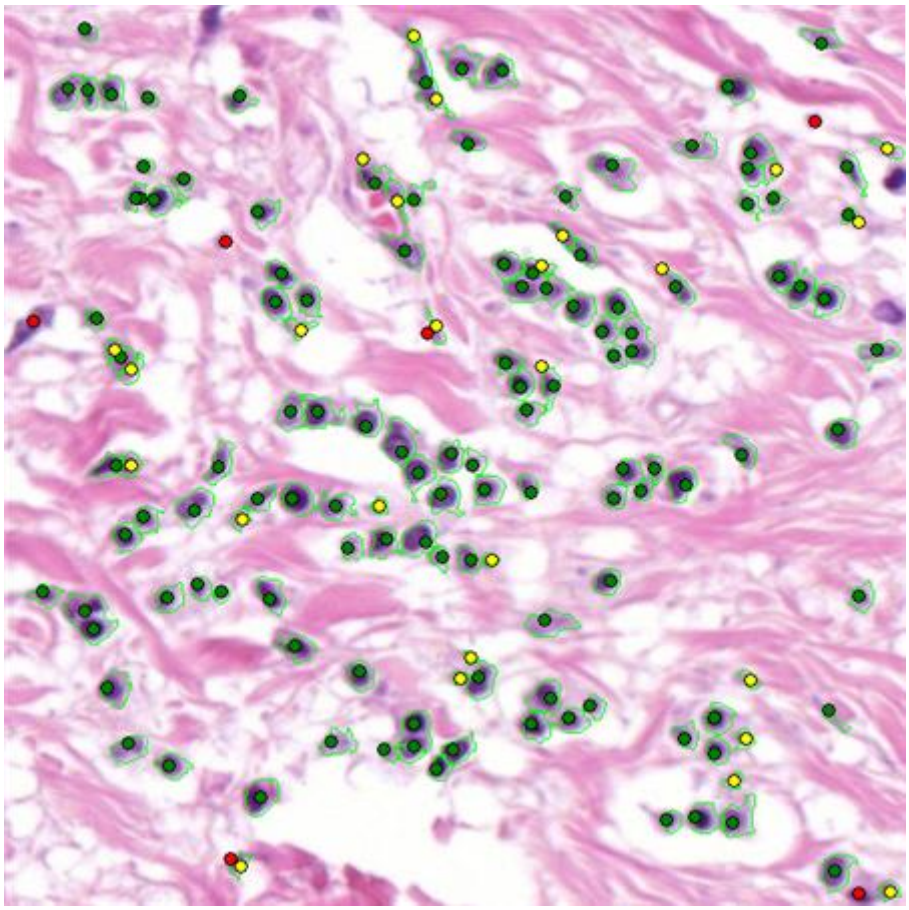
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	136	136	136
true positive count	125	127	127
false negative count	11	9	9
false positive count	10	77	43
precision	0,926	0,623	0,747
recall	0,919	0,934	0,934
conglomerate	0,960	0,992	0,992
time(ms)	546	655	1669

18





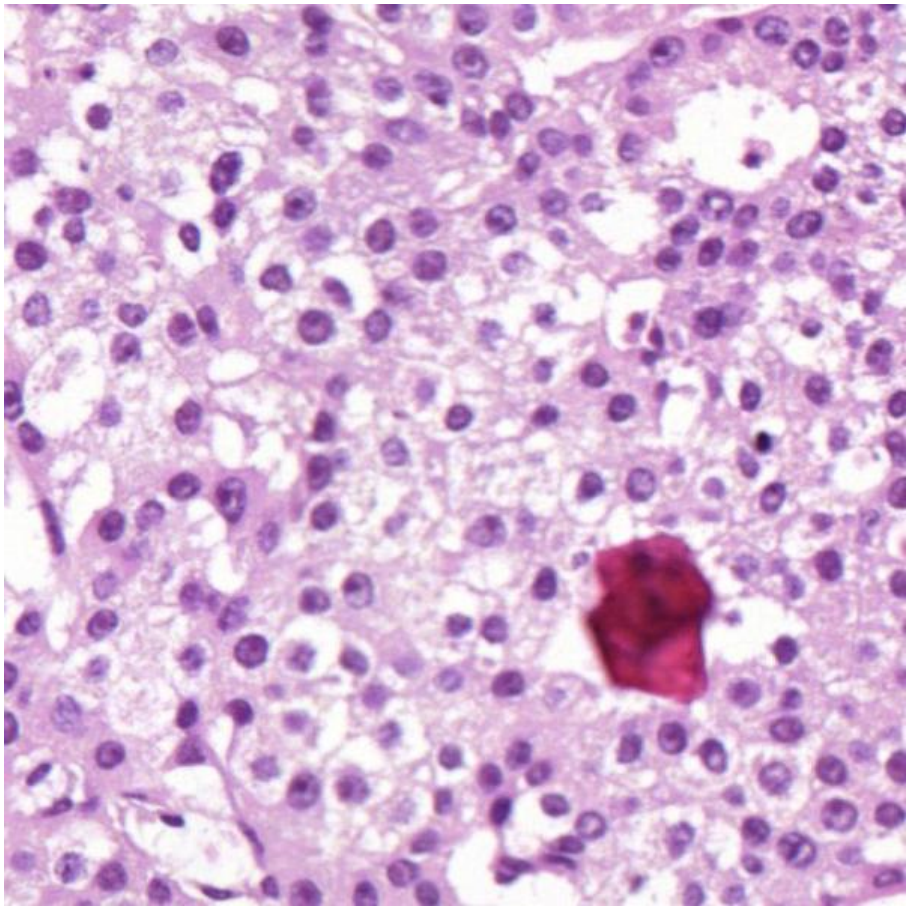
Wienert

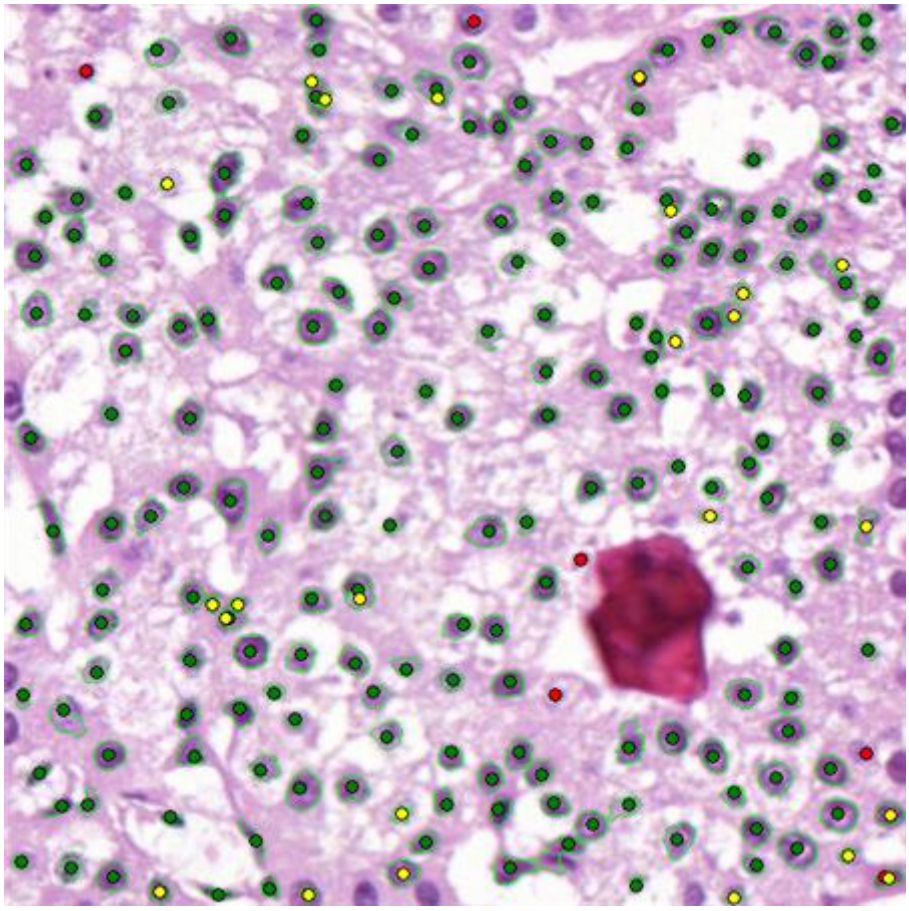


Al-Kofahi (optimized)

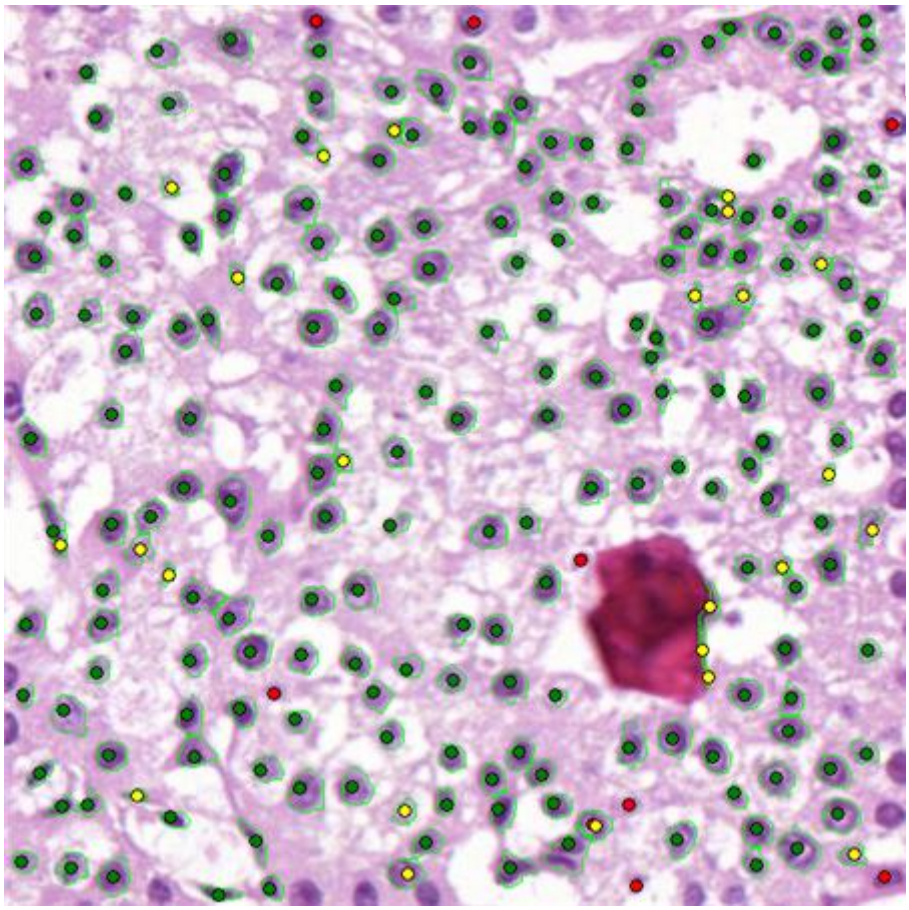
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	126	126	126
true positive count	106	120	120
false negative count	20	6	6
false positive count	8	63	26
precision	0,930	0,656	0,822
recall	0,841	0,952	0,952
conglomerate	0,962	0,950	0,967
time(ms)	561	608	1528

19





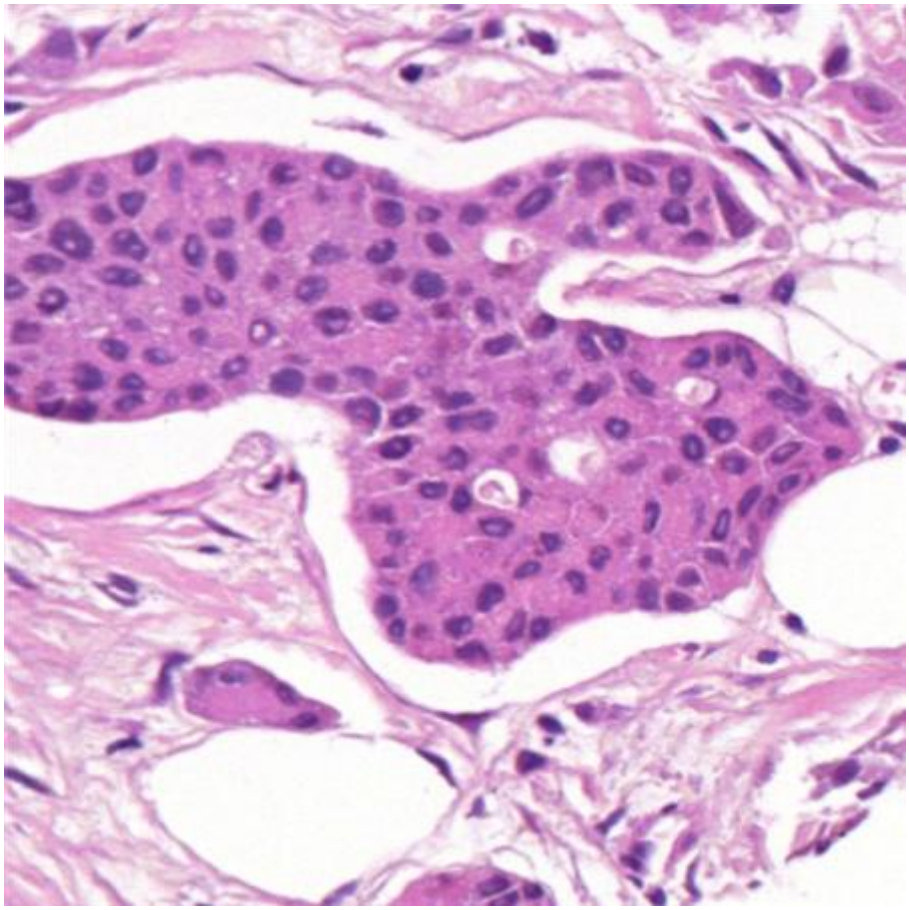
Wienert

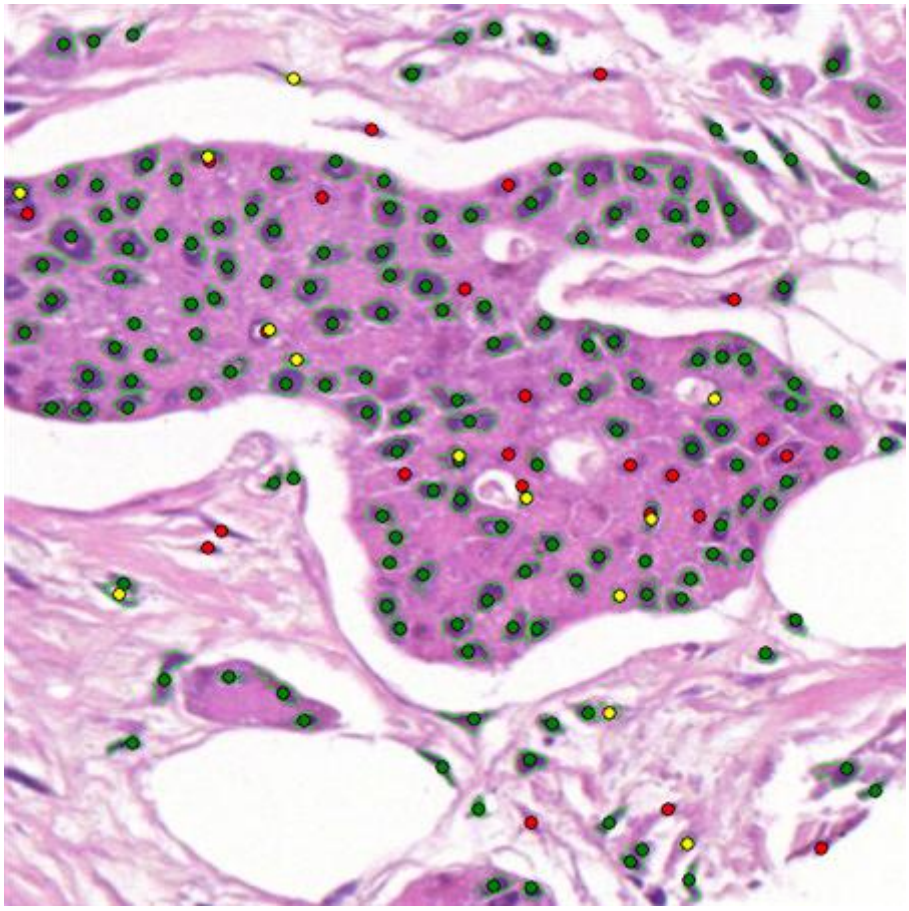


Al-Kofahi (optimized)

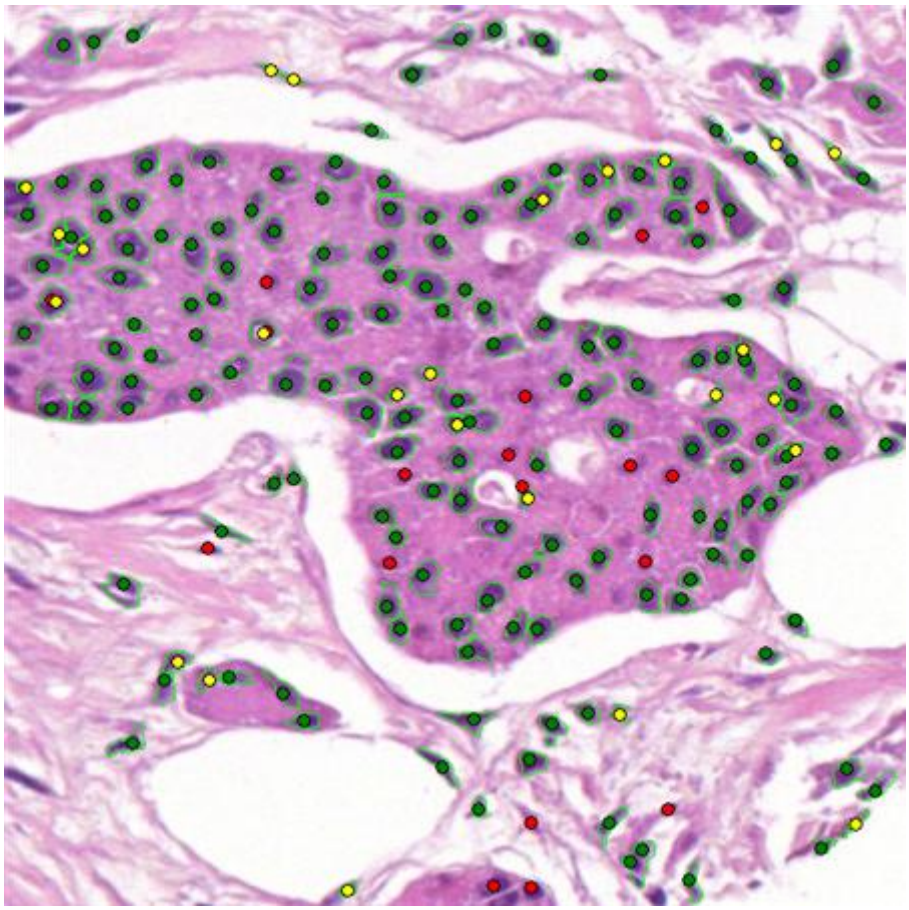
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	204	204	204
true positive count	197	195	195
false negative count	7	9	9
false positive count	24	30	24
precision	0,891	0,867	0,890
recall	0,966	0,956	0,956
conglomerate	0,980	1,000	1,000
time(ms)	780	686	1872

20





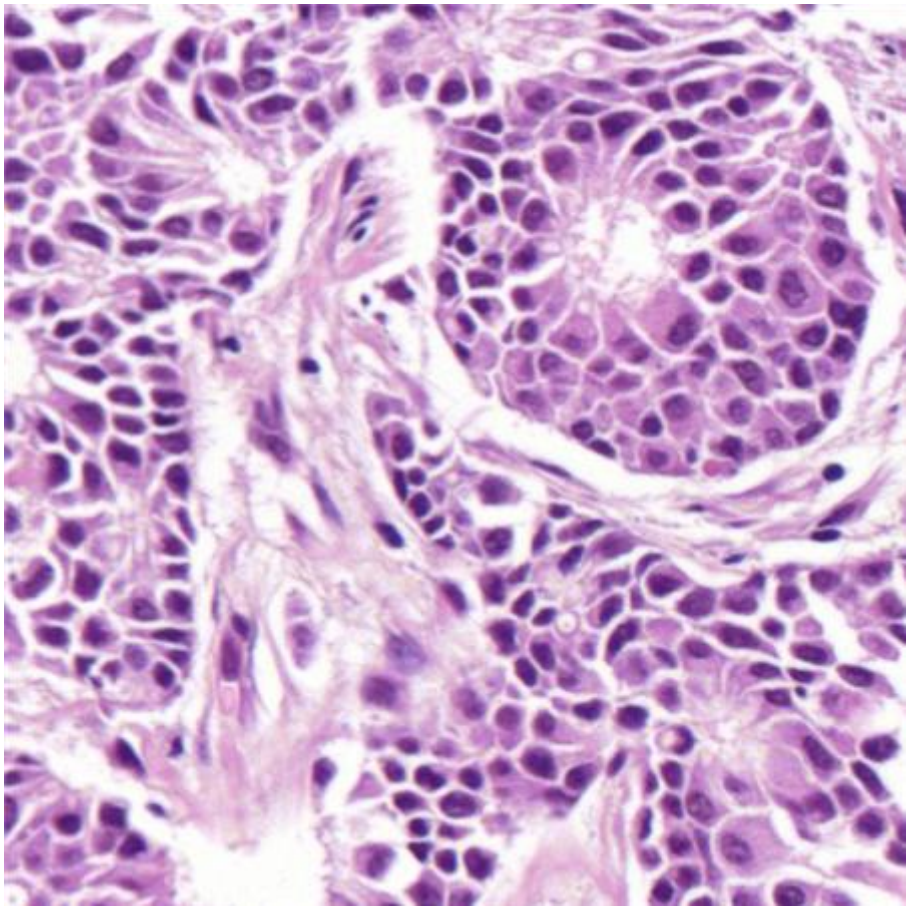
Wienert

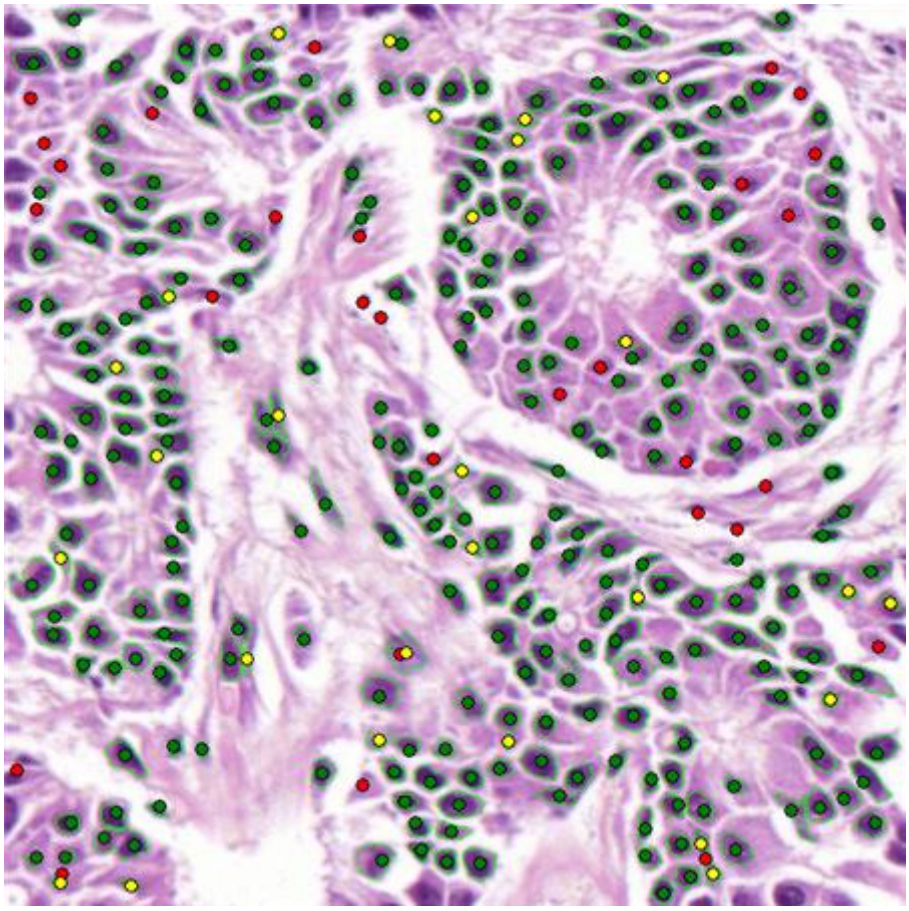


Al-Kofahi (optimized)

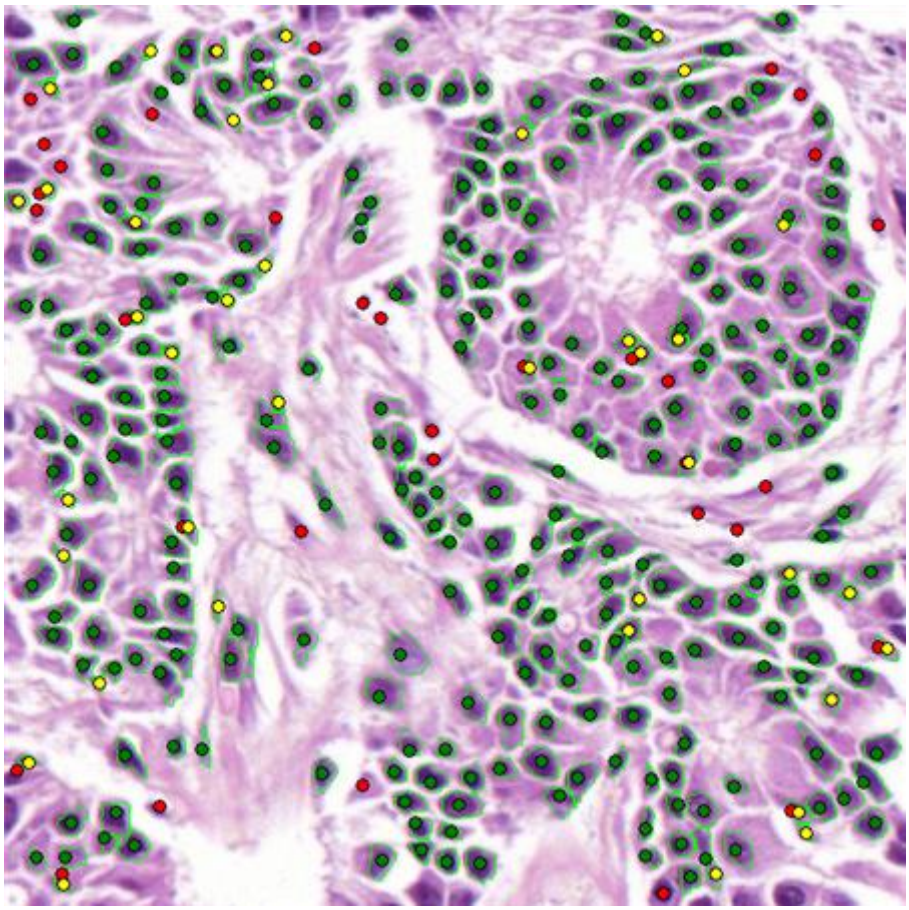
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	180	180	180
true positive count	156	161	162
false negative count	24	19	18
false positive count	13	33	25
precision	0,923	0,830	0,866
recall	0,867	0,894	0,900
conglomerate	0,987	0,981	0,981
time(ms)	624	717	1731

21





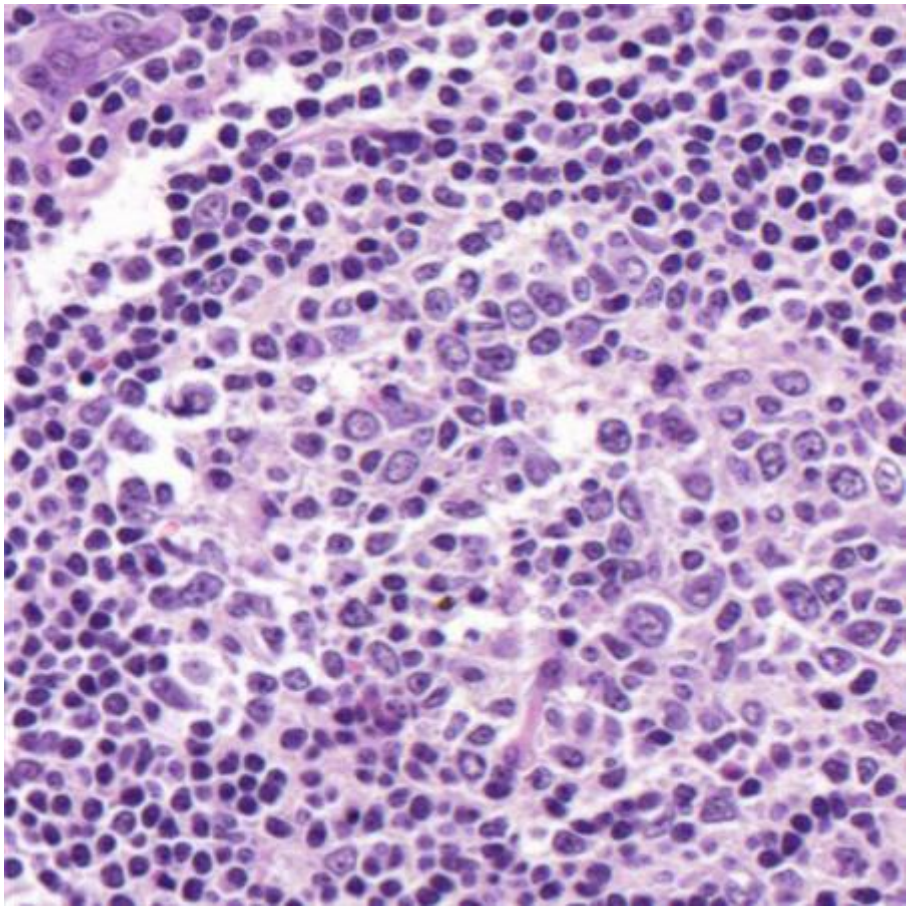
Wienert

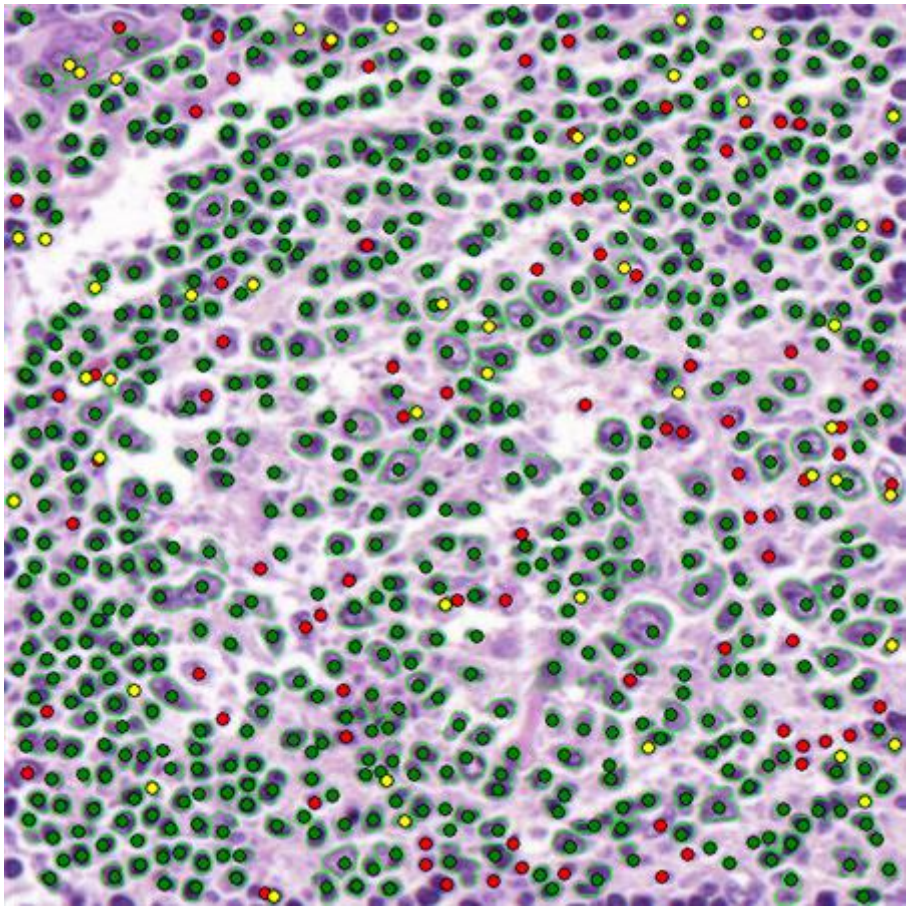


Al-Kofahi (optimized)

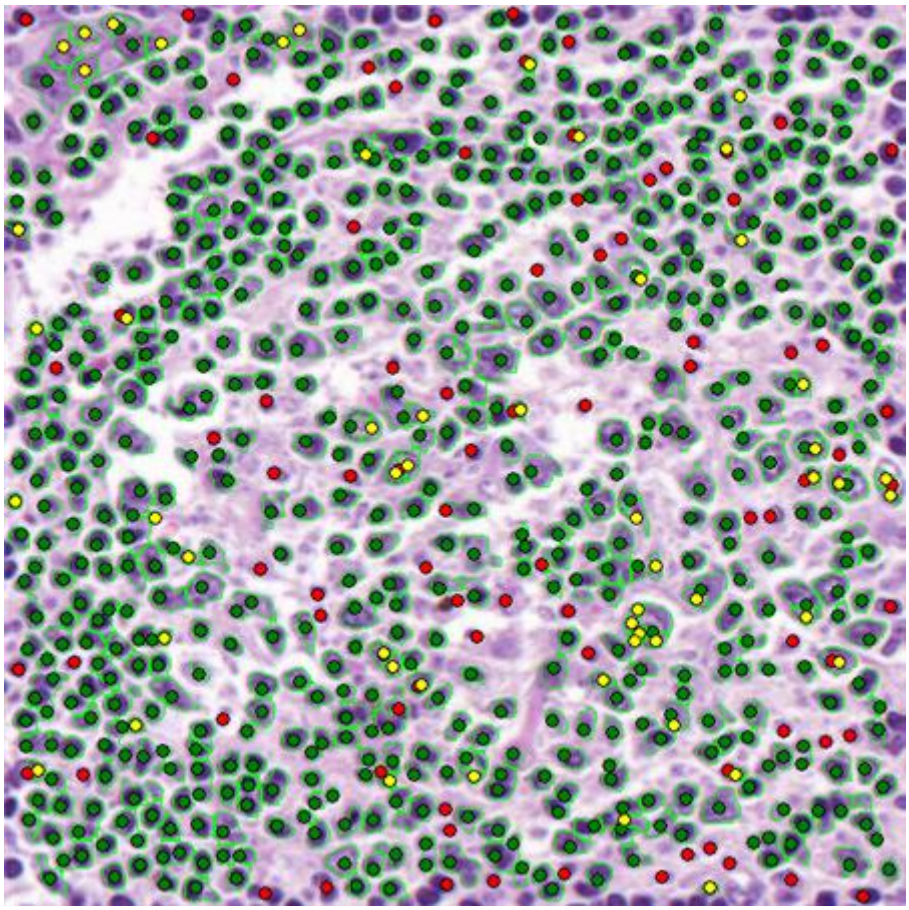
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	311	311	311
true positive count	282	277	277
false negative count	29	34	34
false positive count	27	91	39
precision	0,913	0,753	0,877
recall	0,907	0,891	0,891
conglomerate	0,968	0,975	0,978
time(ms)	811	733	2230

22





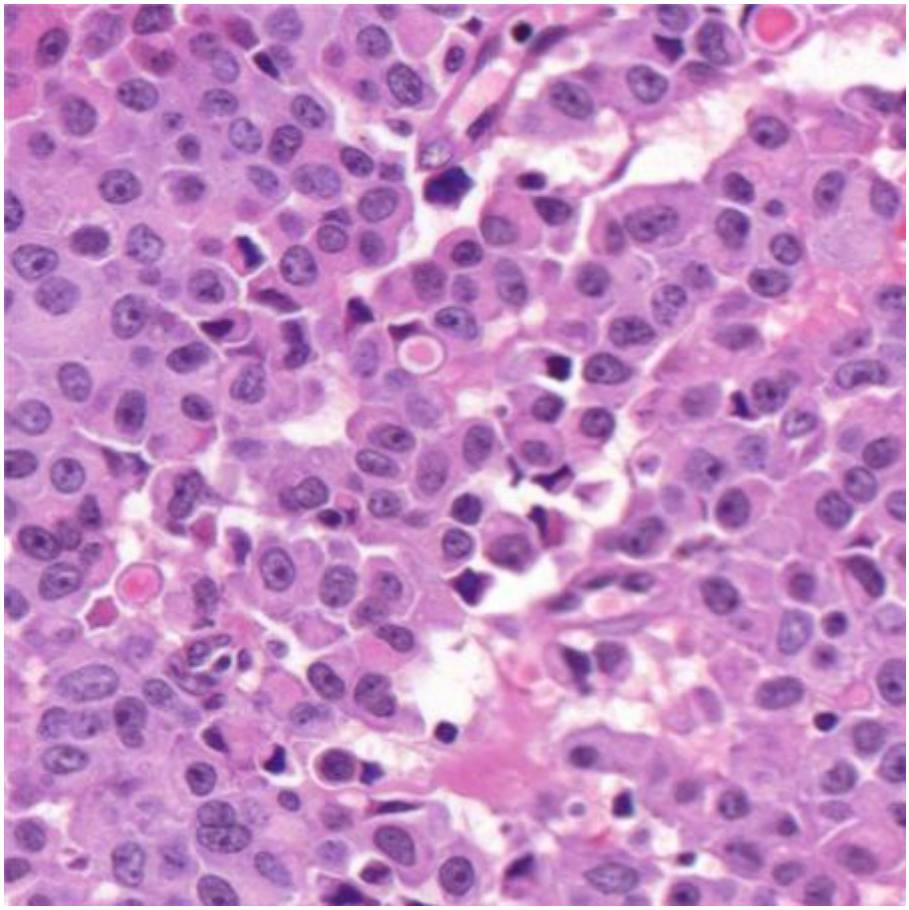
Wienert

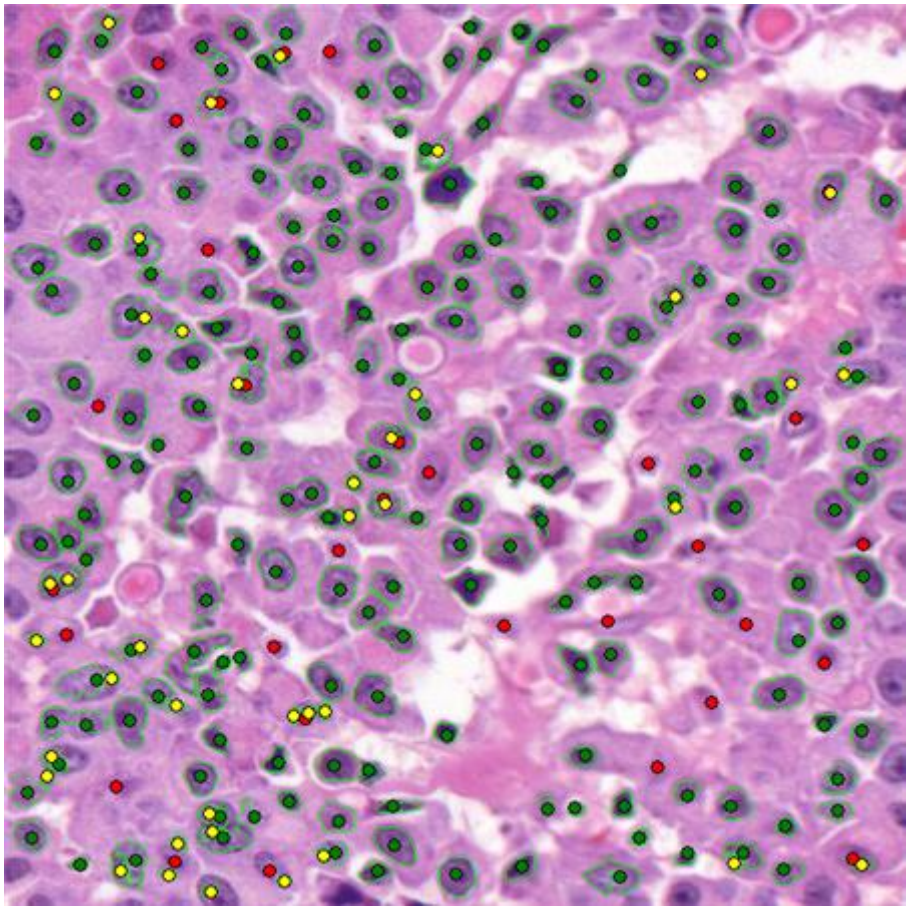


Al-Kofahi (optimized)

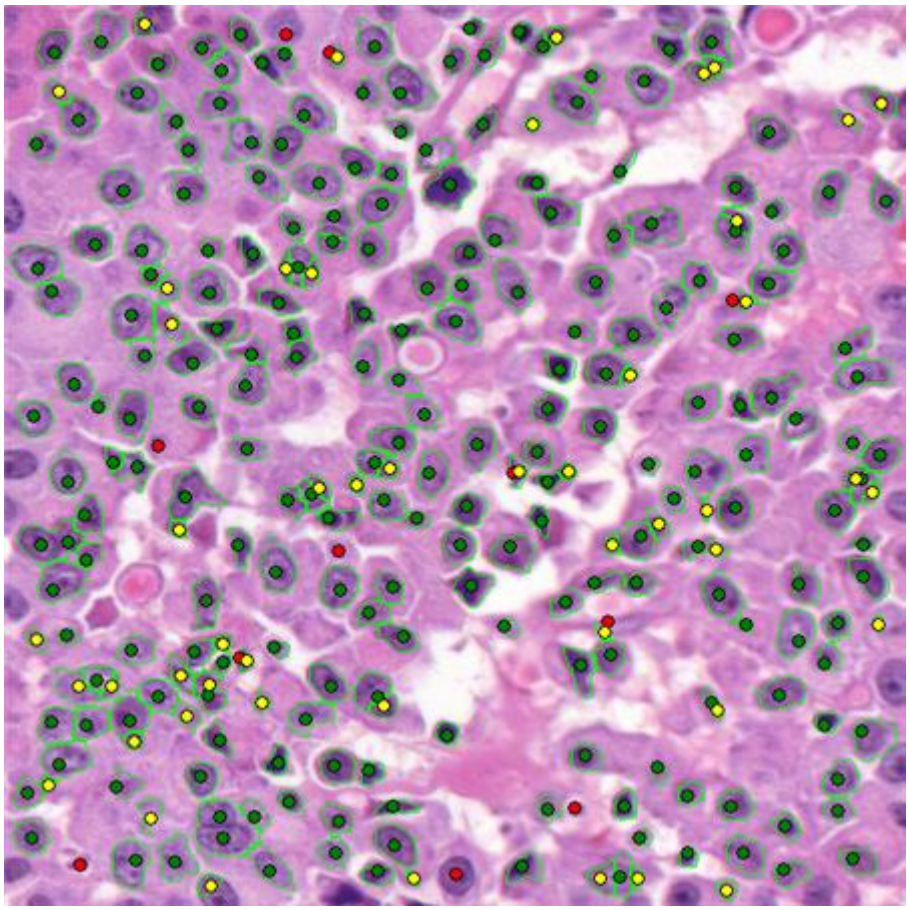
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	688	688	688
true positive count	608	607	603
false negative count	80	81	85
false positive count	47	242	54
precision	0,928	0,715	0,918
recall	0,884	0,882	0,876
conglomerate	0,903	0,913	0,892
time(ms)	1419	702	3042

23





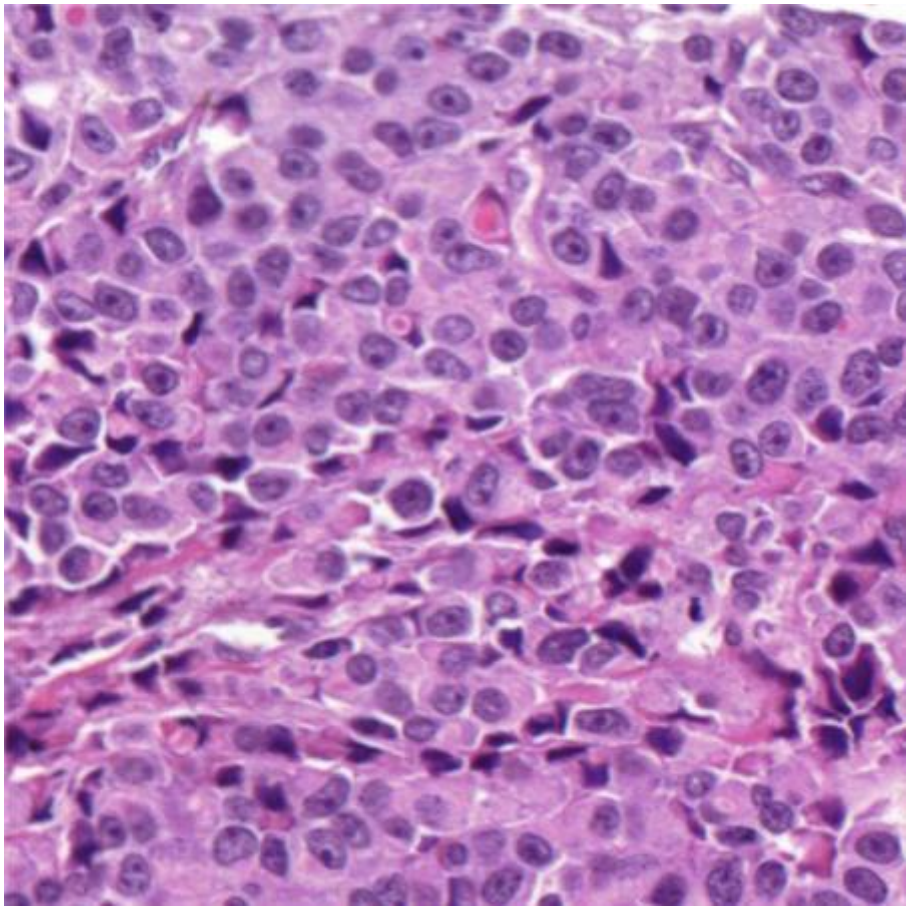
Wienert

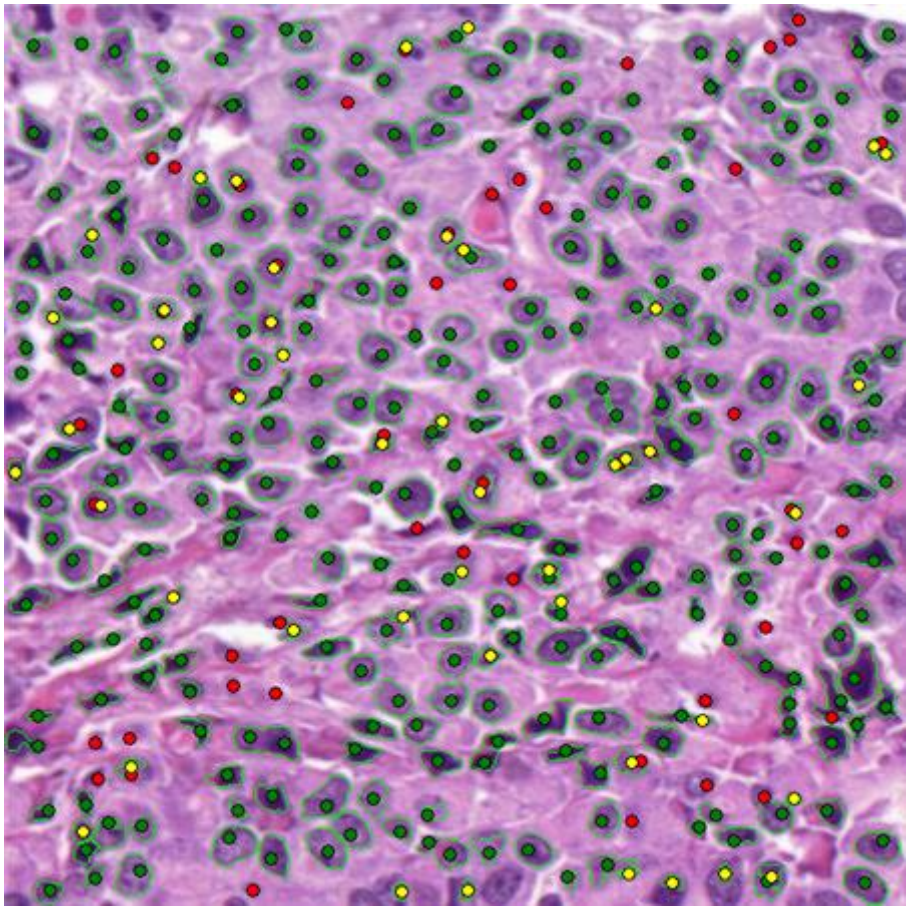


Al-Kofahi (optimized)

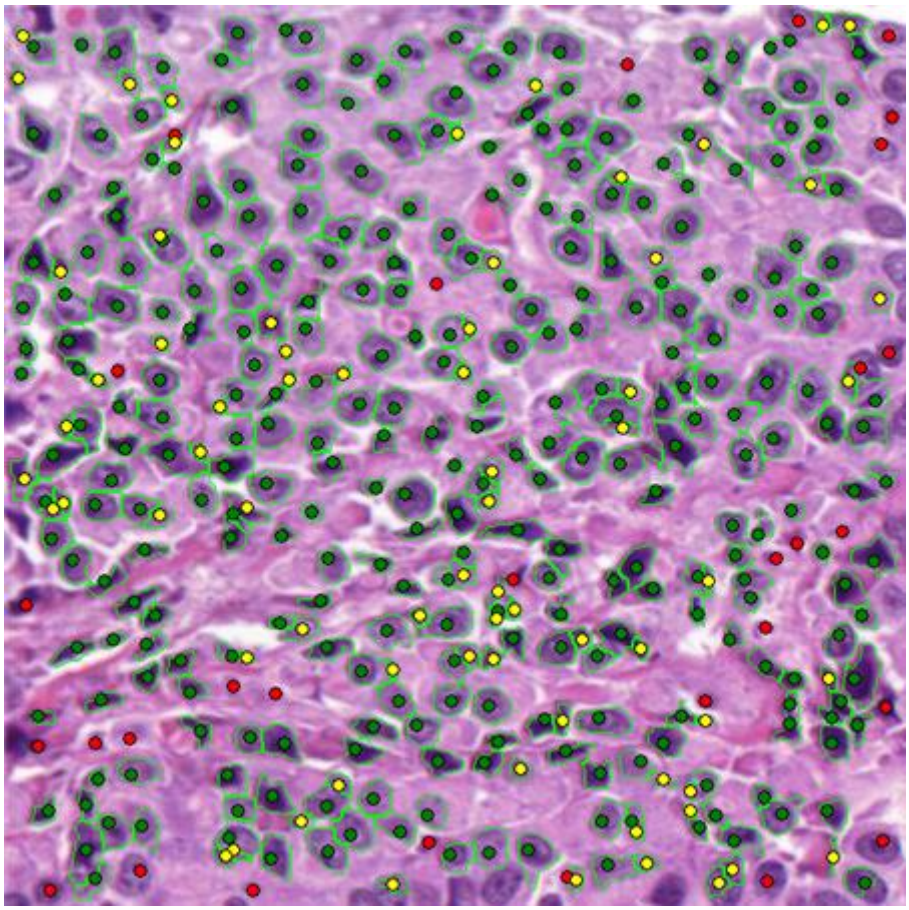
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	233	233	233
true positive count	202	222	222
false negative count	31	11	11
false positive count	41	58	49
precision	0,831	0,793	0,819
recall	0,867	0,953	0,953
conglomerate	0,980	0,977	0,982
time(ms)	904	670	2137

24





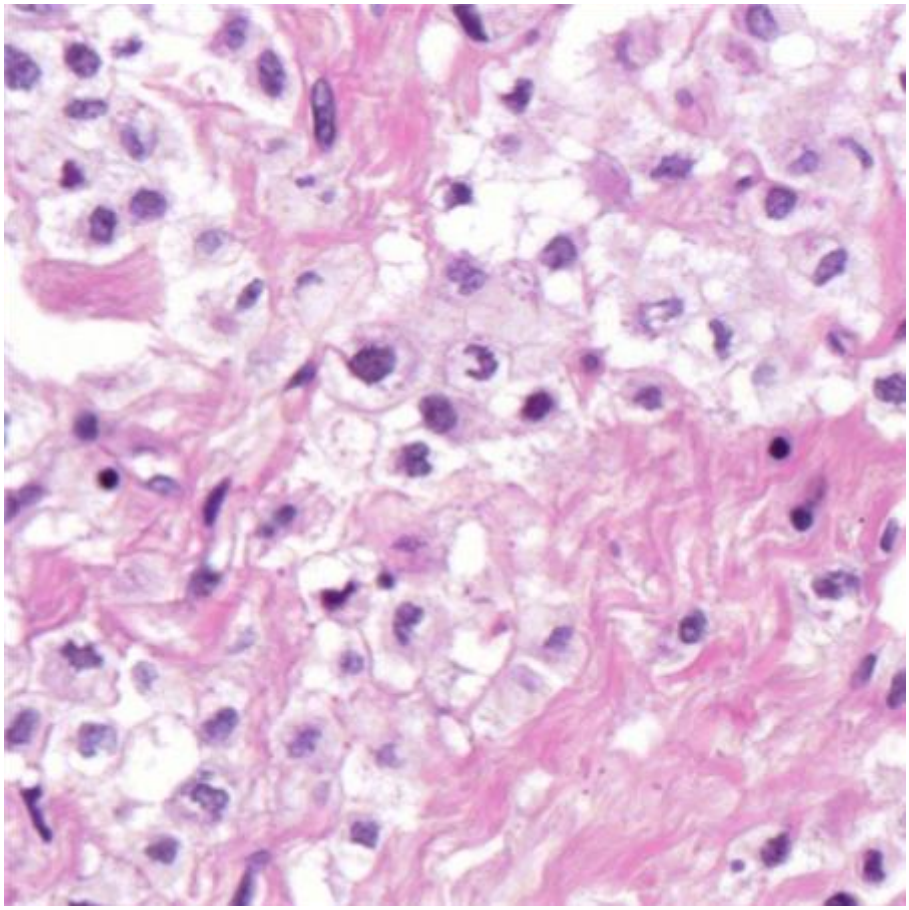
Wienert

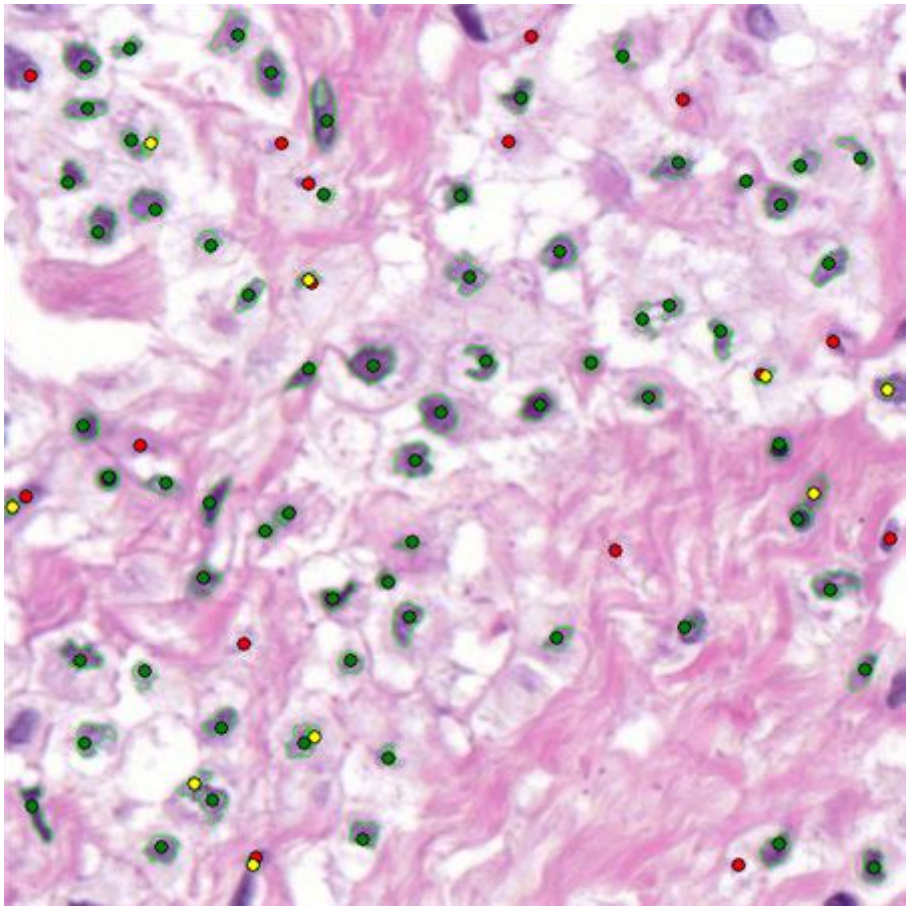


Al-Kofahi (optimized)

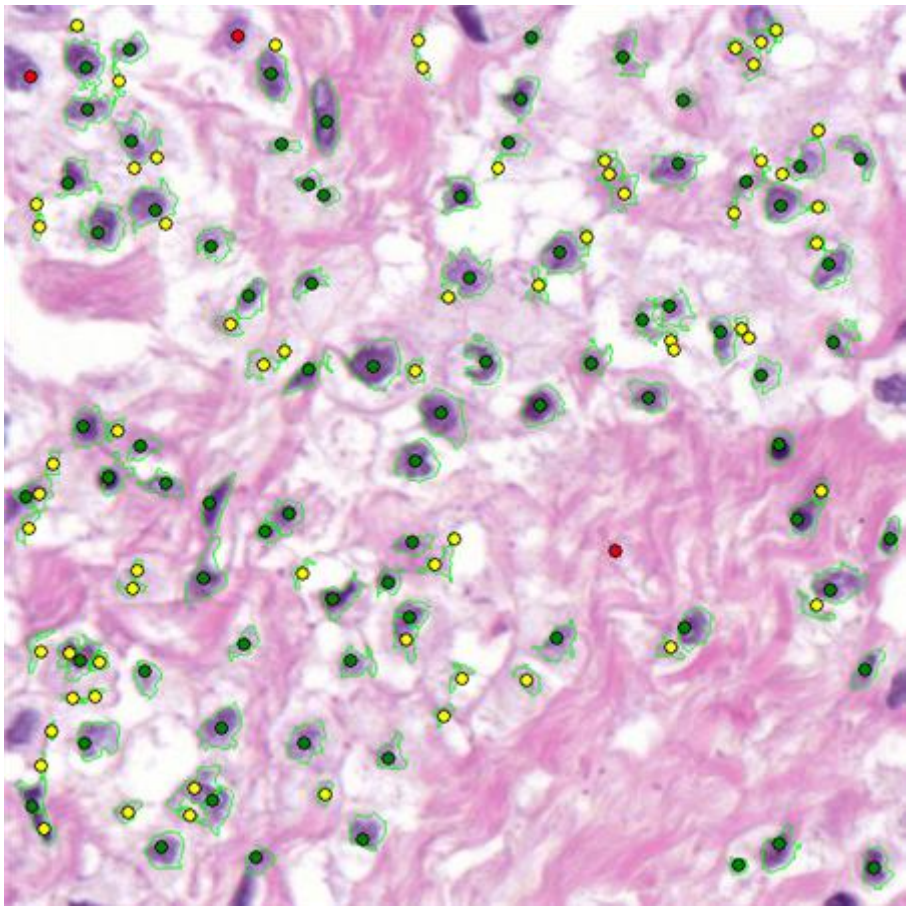
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	329	329	329
true positive count	280	299	299
false negative count	49	30	30
false positive count	46	101	69
precision	0,859	0,748	0,813
recall	0,851	0,909	0,909
conglomerate	0,971	0,967	0,987
time(ms)	1076	702	2449

25





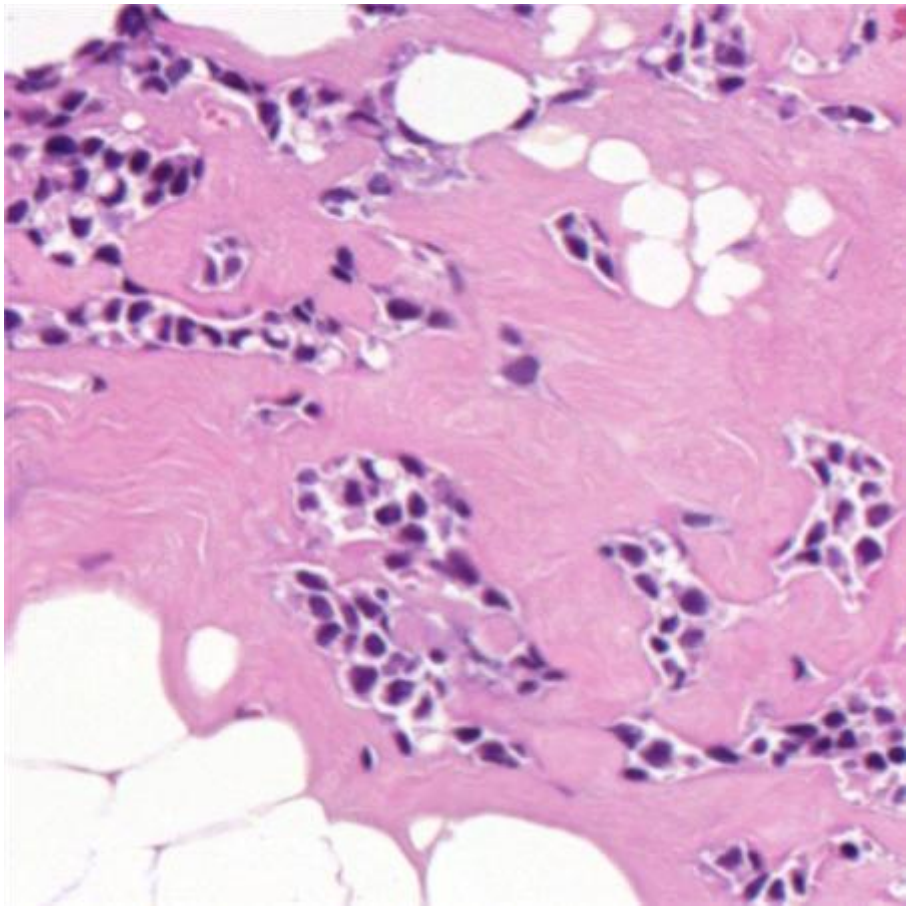
Wienert

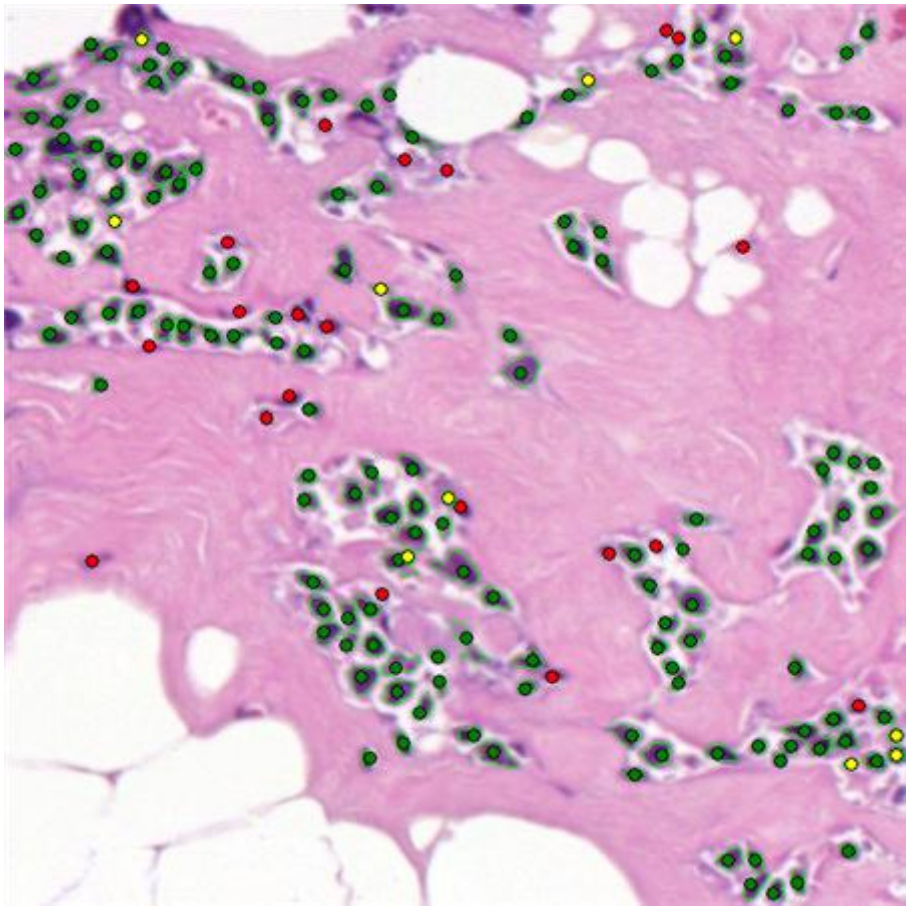


Al-Kofahi (optimized)

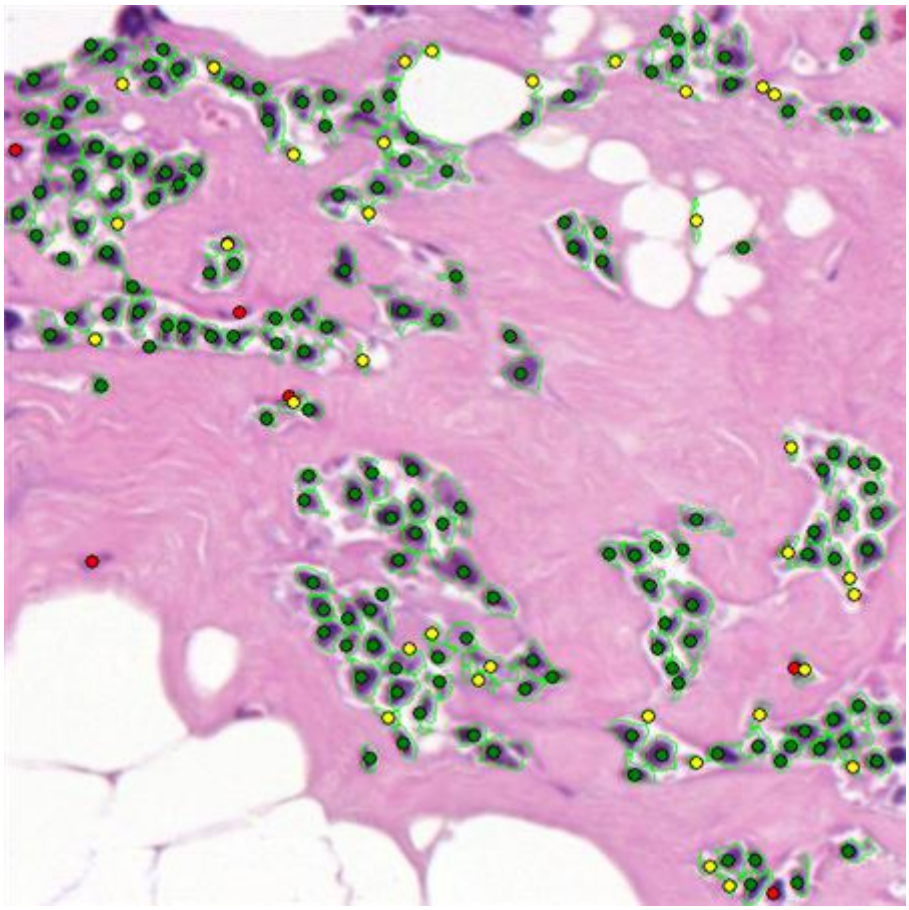
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	81	81	81
true positive count	65	78	78
false negative count	16	3	3
false positive count	9	98	63
precision	0,878	0,443	0,553
recall	0,802	0,963	0,963
conglomerate	1,000	1,000	1,000
time(ms)	530	670	1653

26





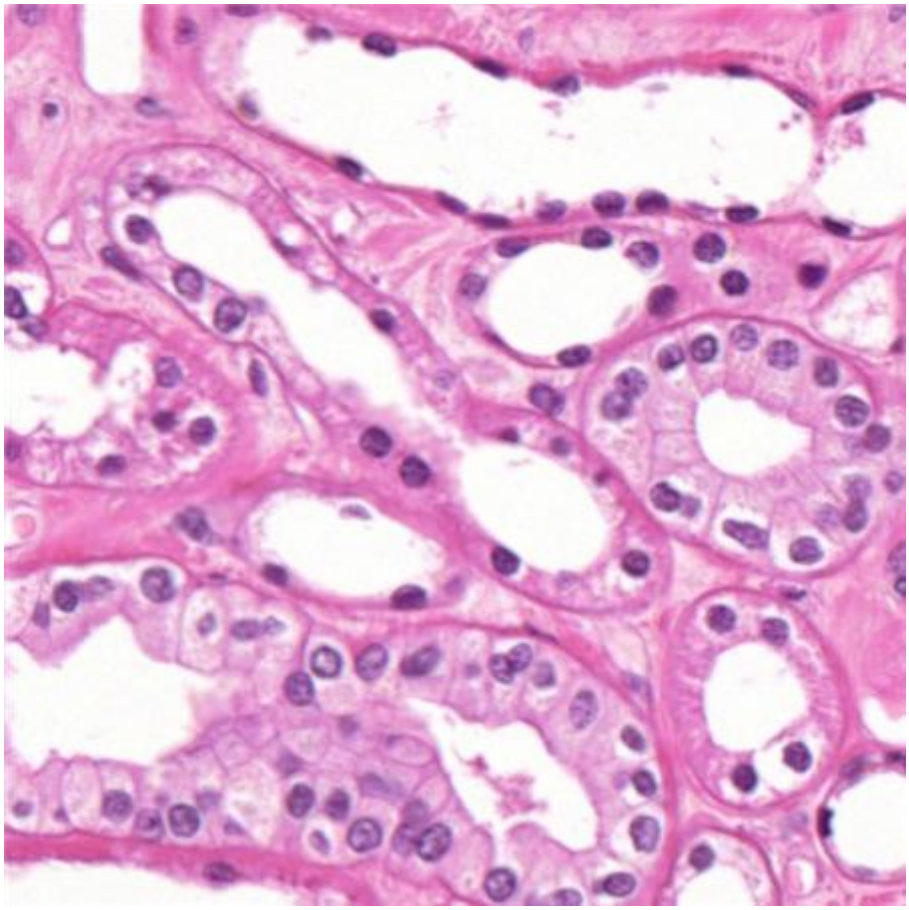
Wienert

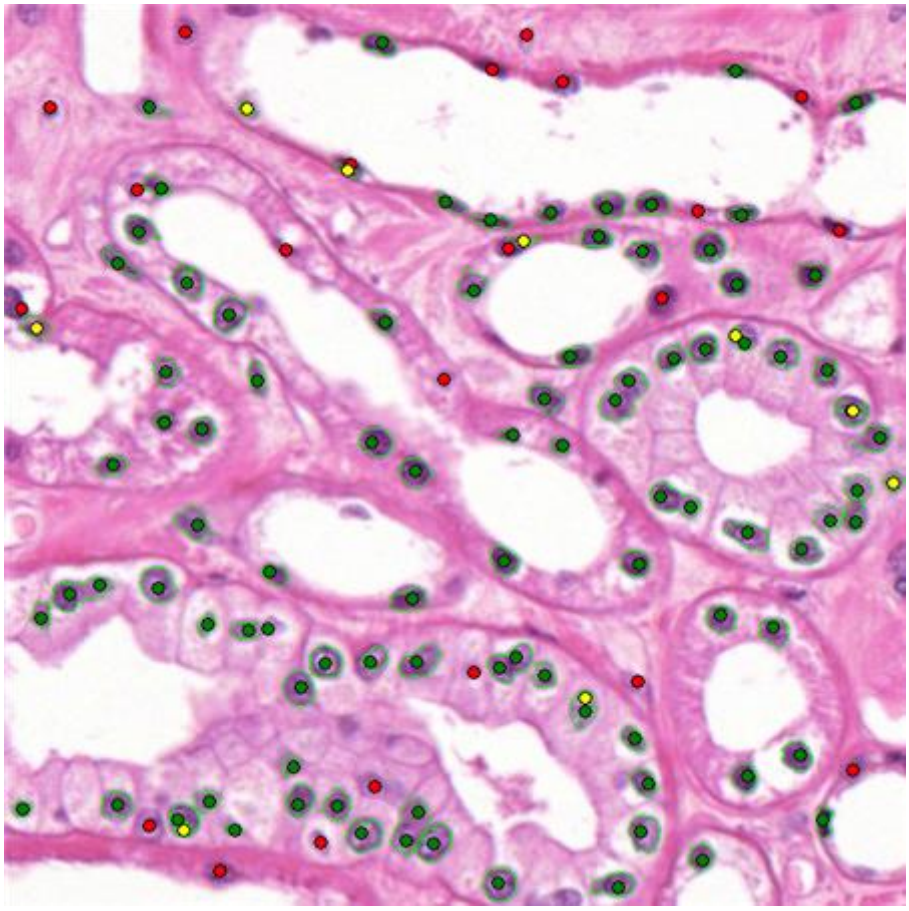


Al-Kofahi (optimized)

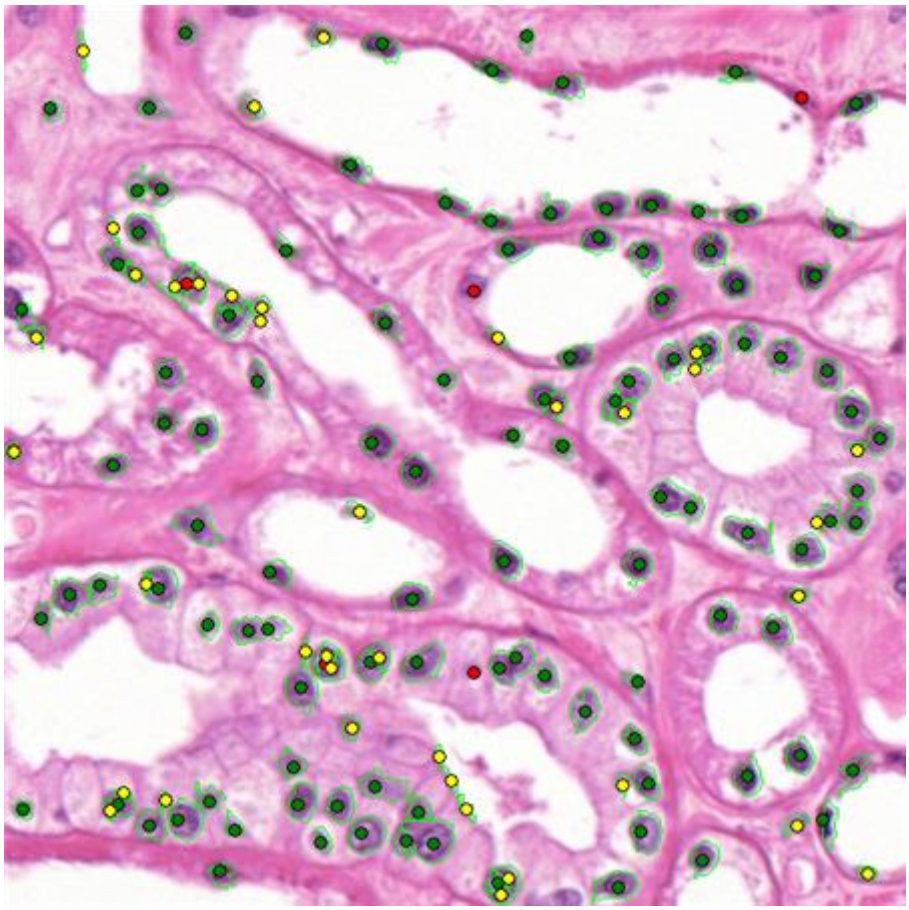
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	169	169	169
true positive count	148	162	162
false negative count	21	7	7
false positive count	10	128	34
precision	0,937	0,559	0,827
recall	0,876	0,959	0,959
conglomerate	0,939	0,944	0,938
time(ms)	655	670	1684

27





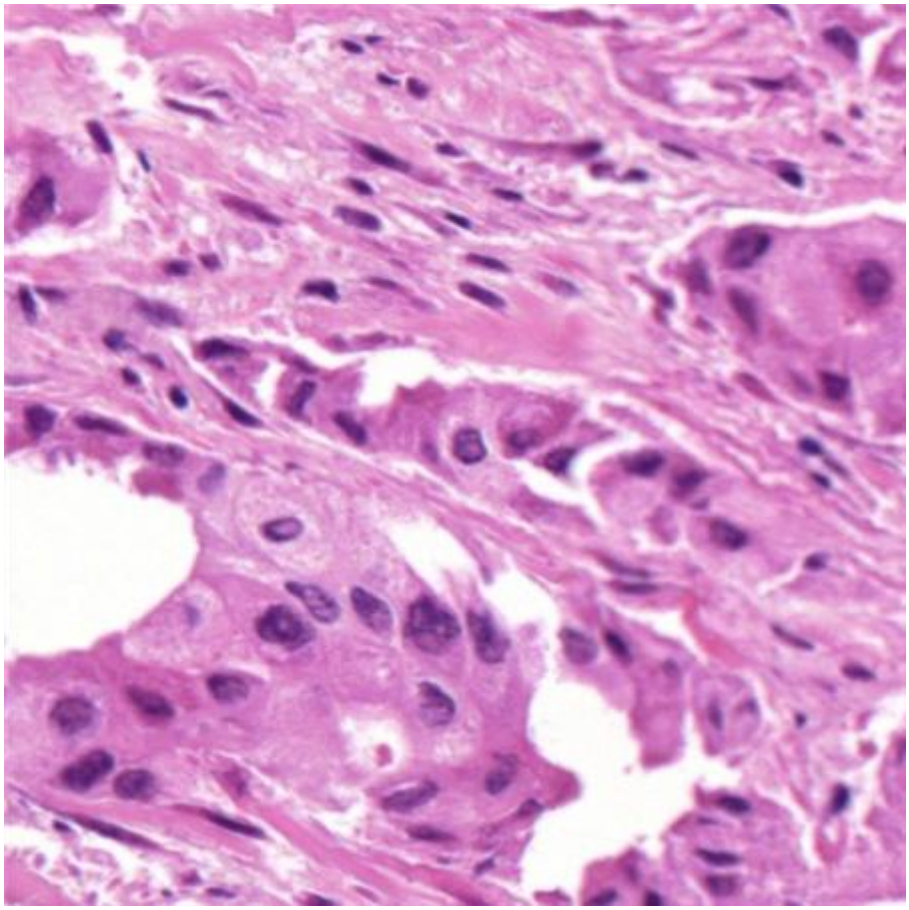
Wienert

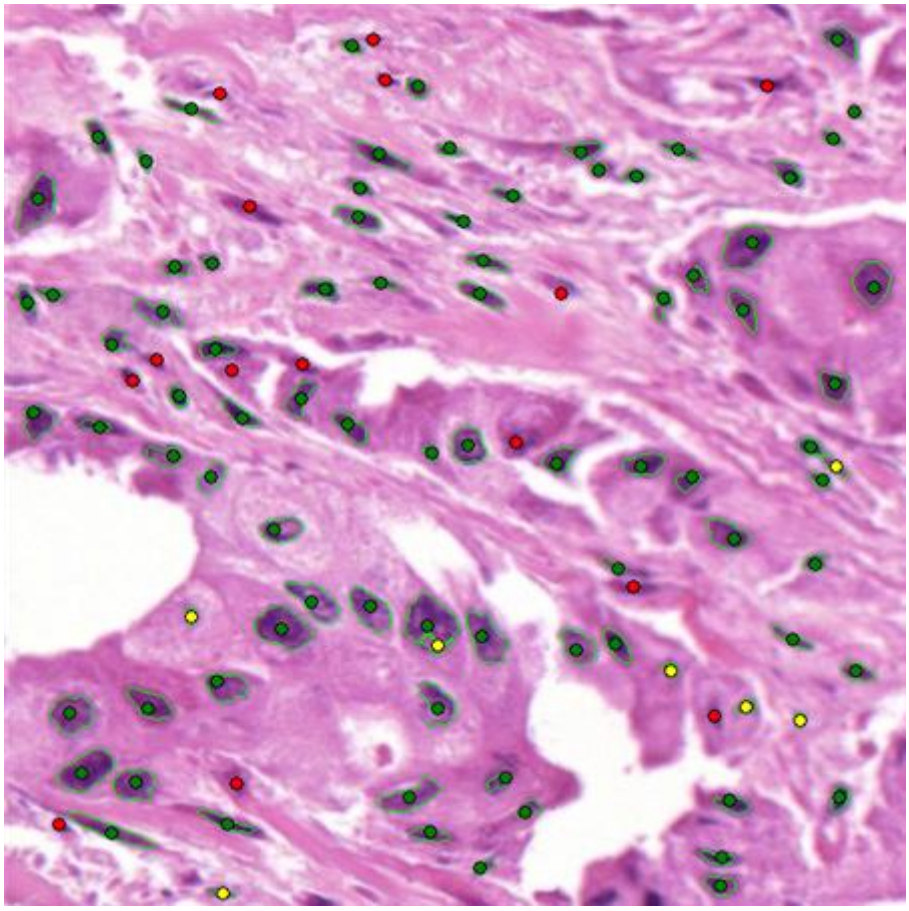


Al-Kofahi (optimized)

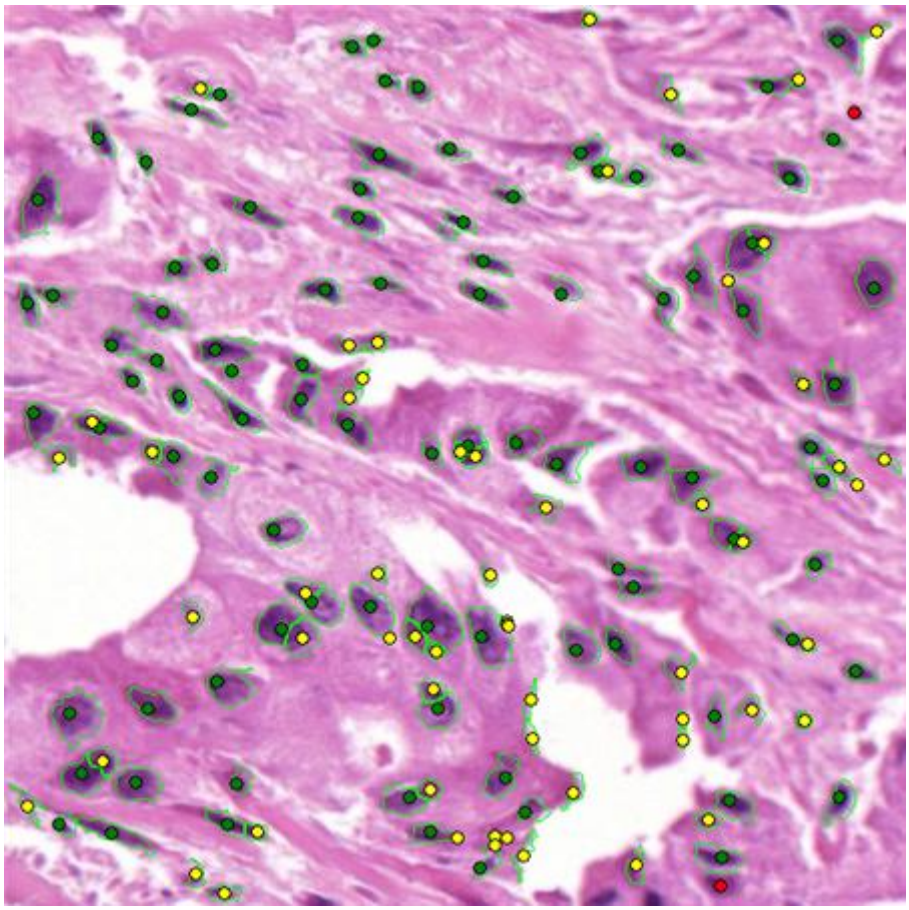
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	114	114	114
true positive count	91	109	109
false negative count	23	5	5
false positive count	9	62	38
precision	0,910	0,637	0,741
recall	0,798	0,956	0,956
conglomerate	0,934	0,972	0,991
time(ms)	577	624	1544

28





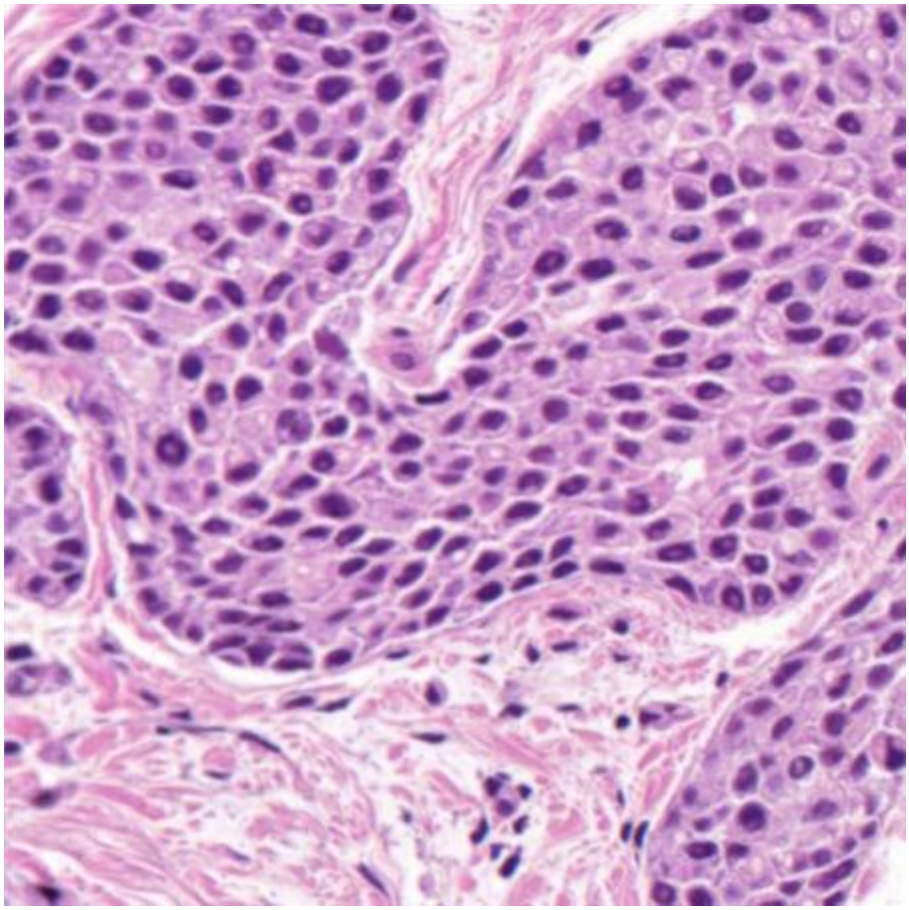
Wienert

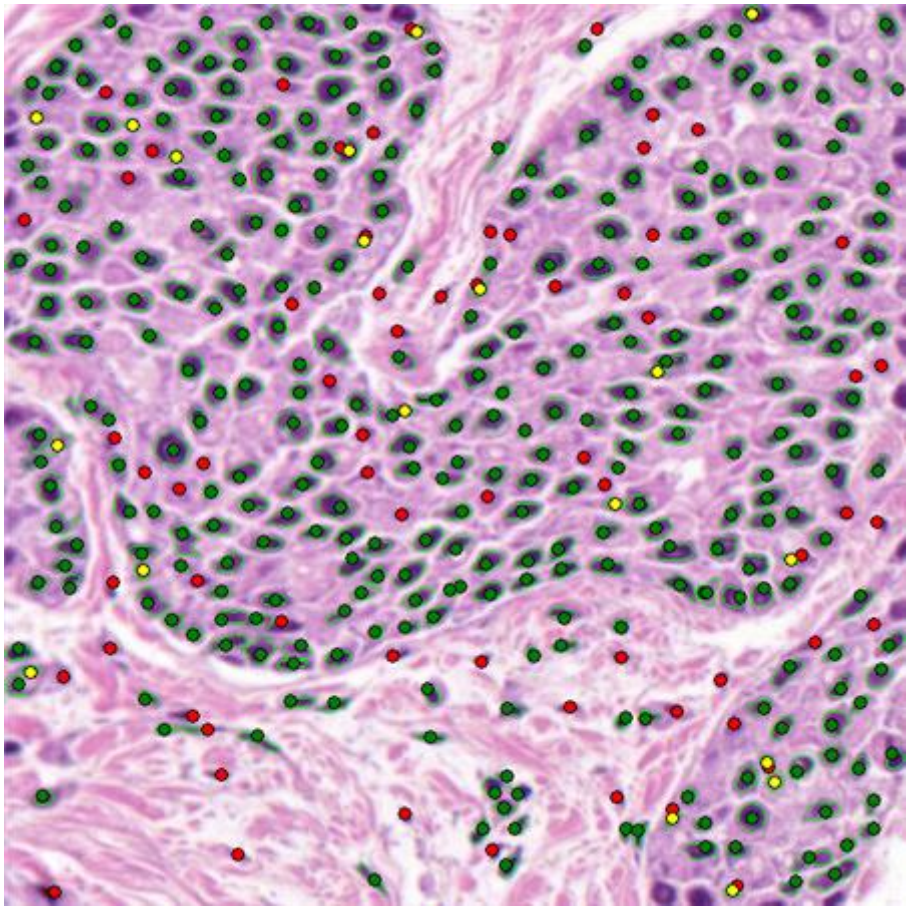


Al-Kofahi (optimized)

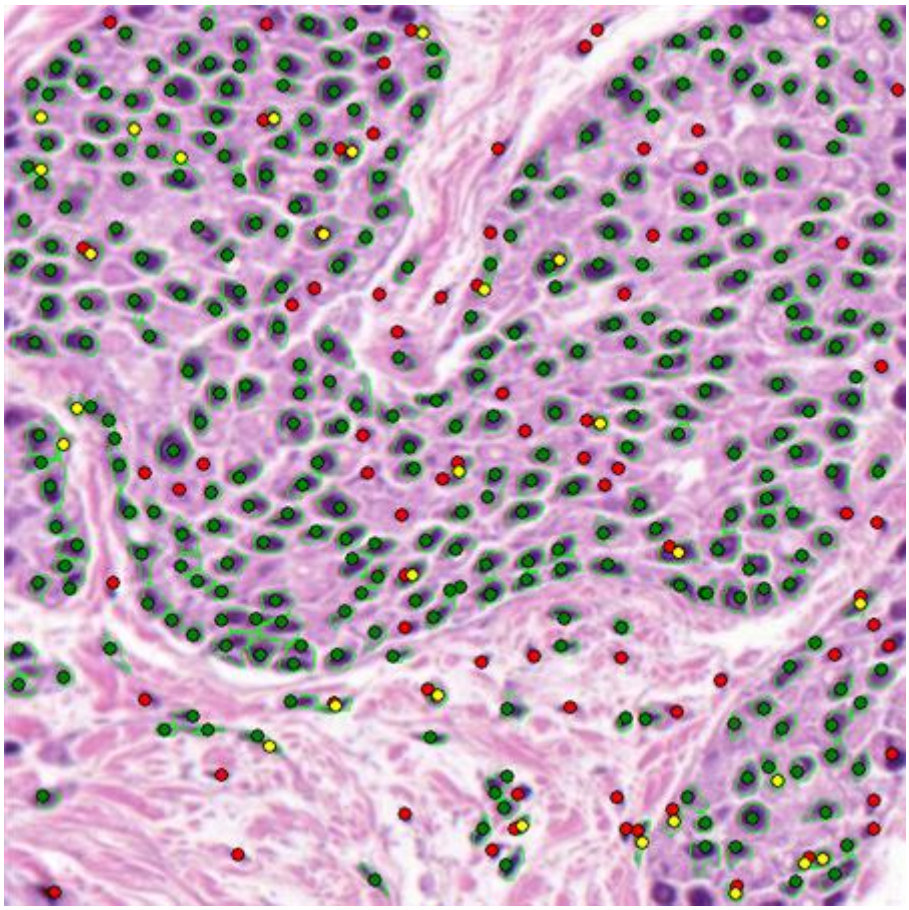
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	97	97	688
true positive count	82	95	603
false negative count	15	2	85
false positive count	7	76	54
precision	0,921	0,556	0,918
recall	0,845	0,979	0,876
conglomerate	1,000	0,989	0,989
time(ms)	530	639	1606

29





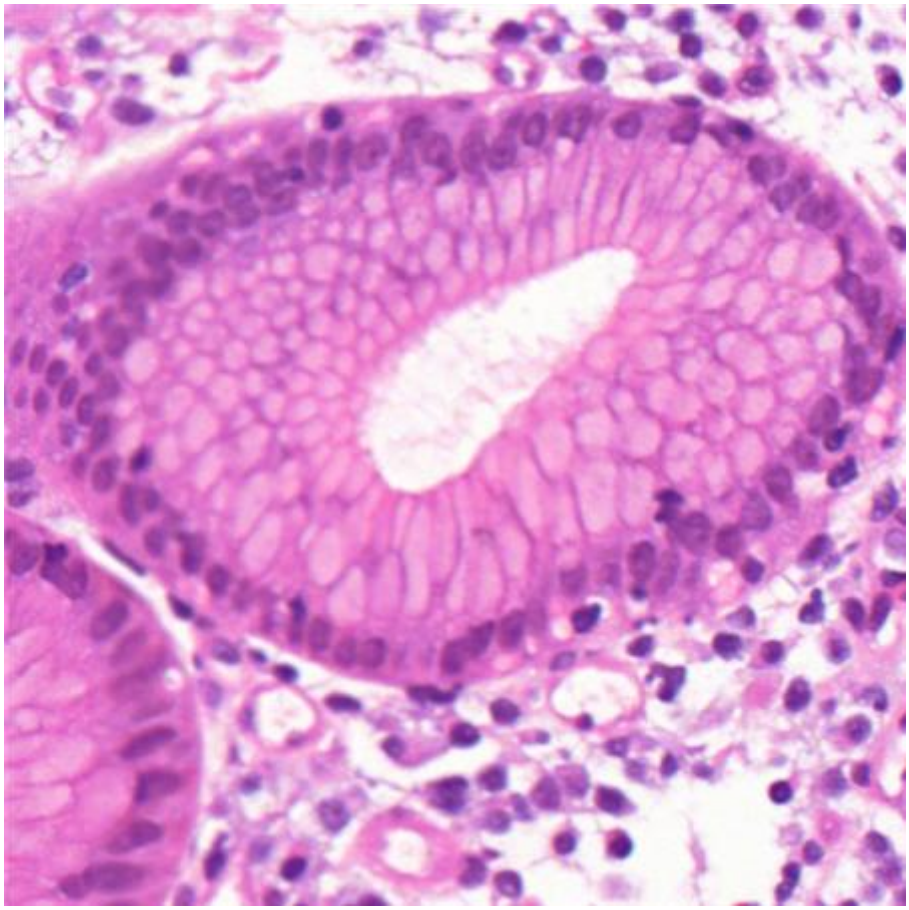
Wienert

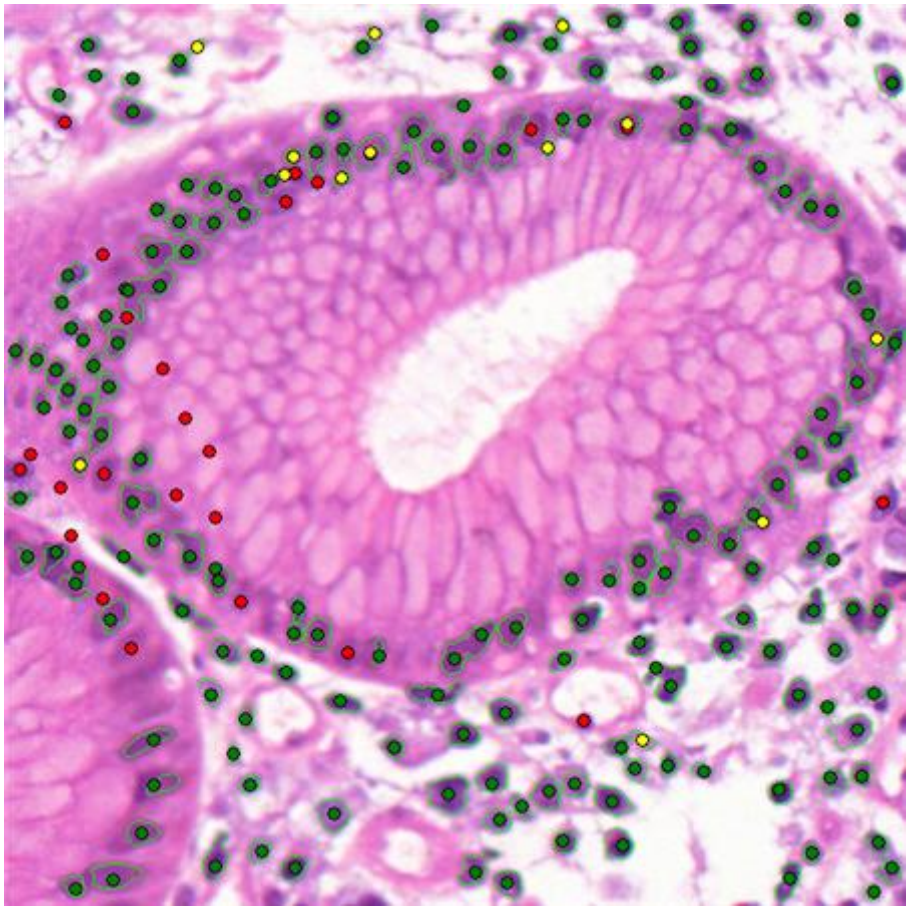


Al-Kofahi (optimized)

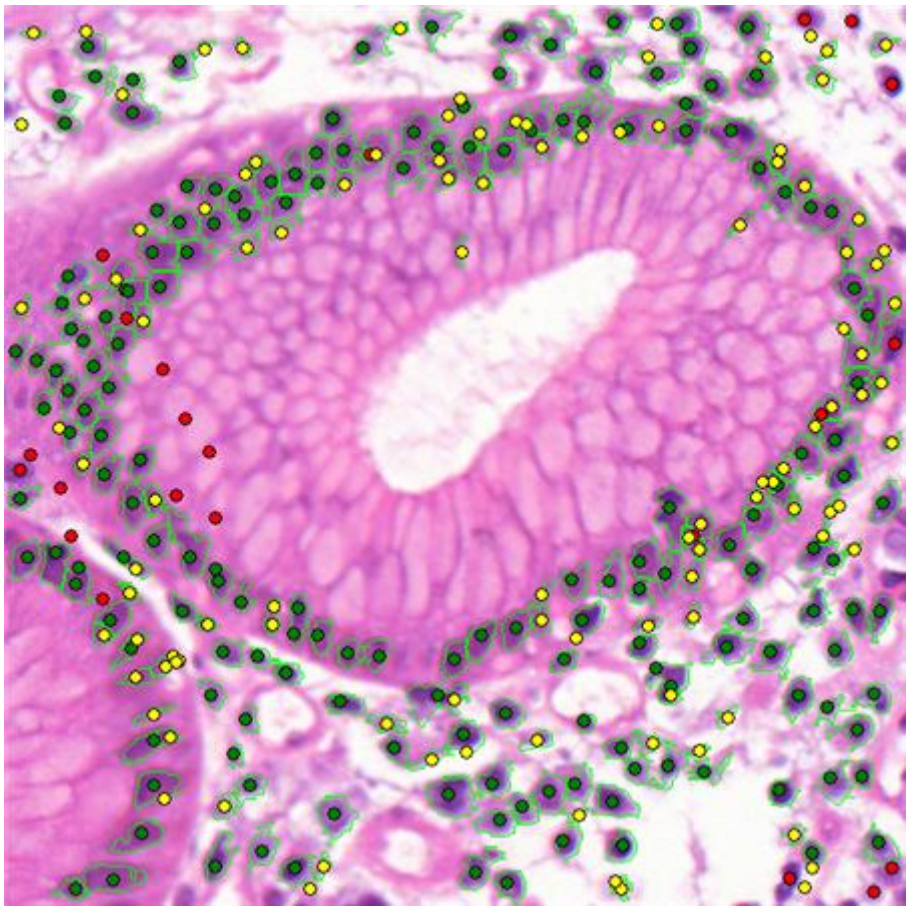
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	392	392	392
true positive count	326	316	316
false negative count	66	76	76
false positive count	19	58	54
precision	0,945	0,845	0,854
recall	0,832	0,806	0,806
conglomerate	0,969	0,959	0,959
time(ms)	717	717	2215

30





Wienert



Al-Kofahi (optimized)

	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	196	196	196
true positive count	171	173	174
false negative count	25	23	22
false positive count	13	249	102
precision	0,929	0,410	0,630
recall	0,872	0,883	0,888
conglomerate	0,928	0,959	0,954
time(ms)	686	670	2184

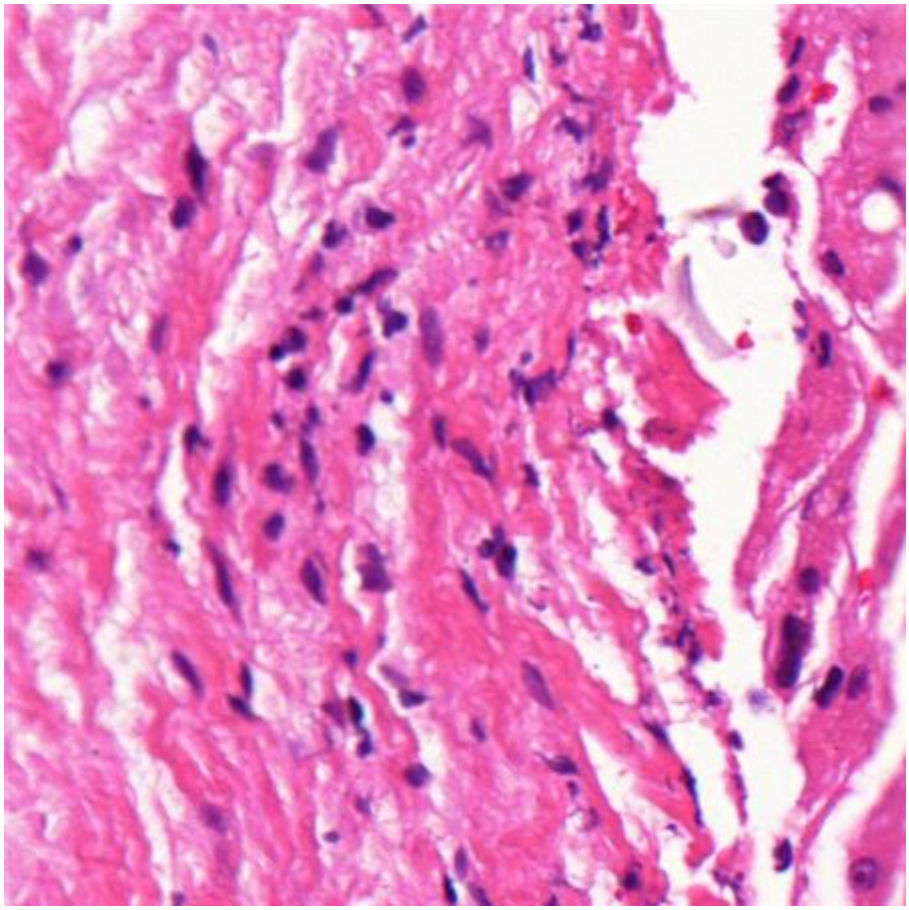
31

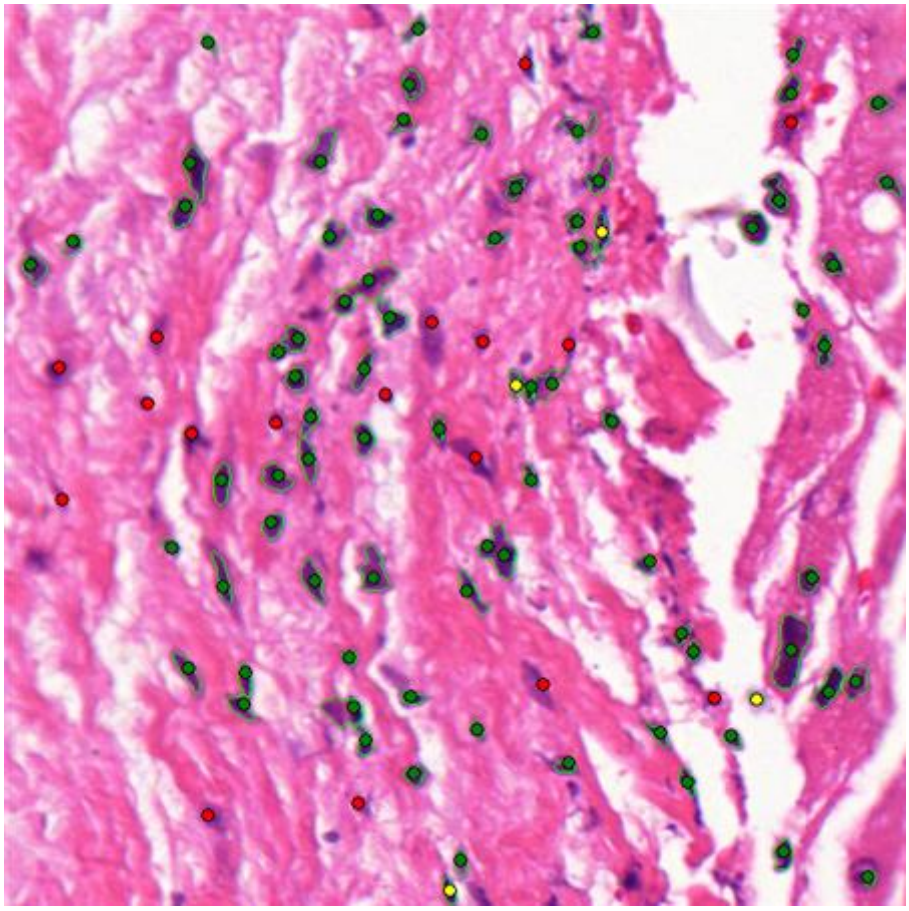
Wienert

Al-Kofahi (optimized)

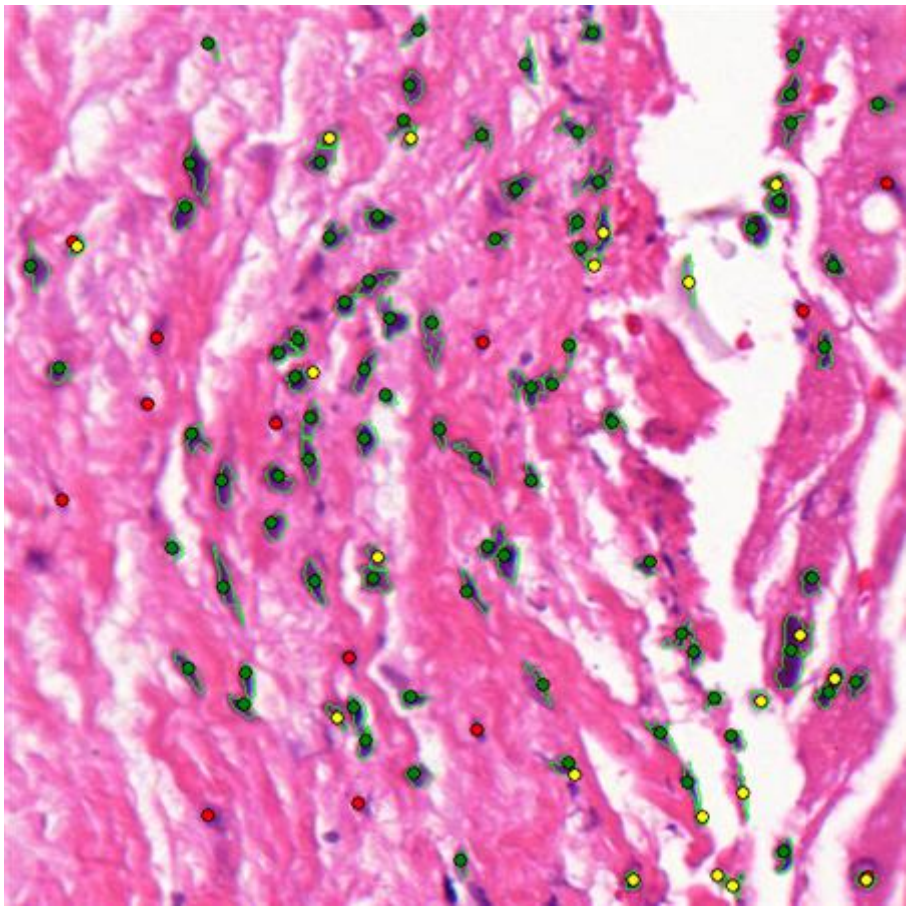
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	115	115	115
true positive count	108	109	108
false negative count	7	6	7
false positive count	7	153	68
precision	0,939	0,416	0,614
recall	0,939	0,948	0,939
conglomerate	0,991	0,963	0,963
time(ms)	670	624	1653

32





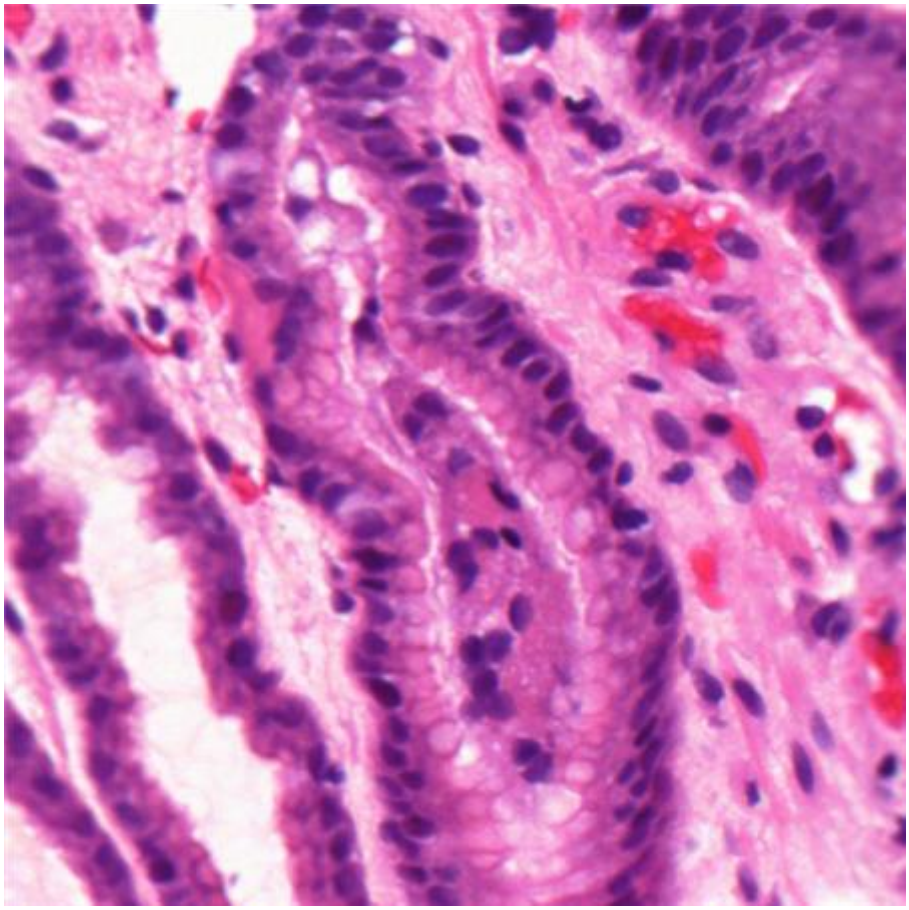
Wienert

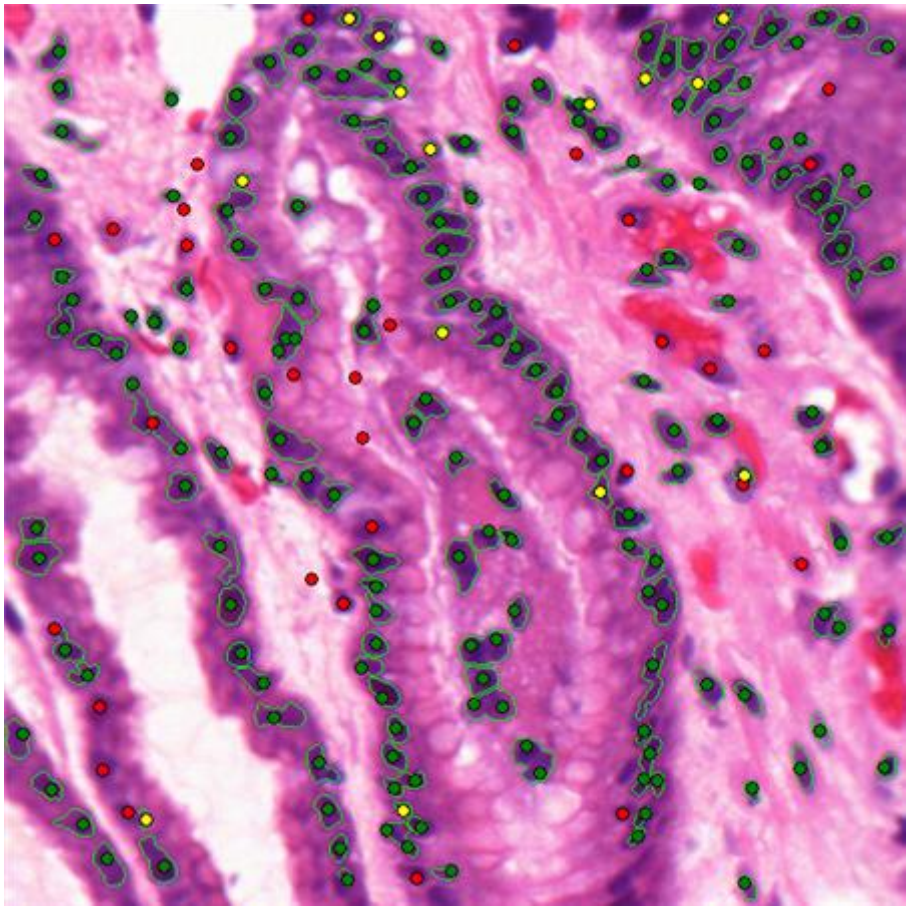


Al-Kofahi (optimized)

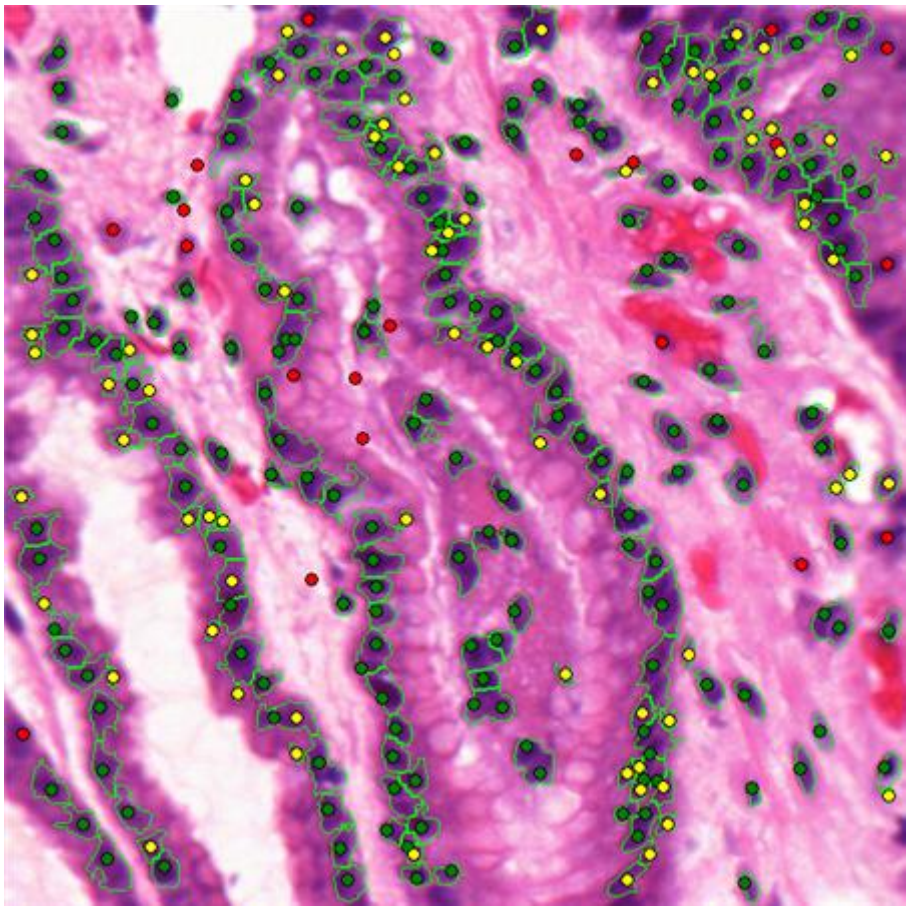
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	93	93	93
true positive count	76	80	80
false negative count	17	13	13
false positive count	4	35	20
precision	0,950	0,696	0,800
recall	0,817	0,860	0,860
conglomerate	0,961	1,000	1,000
time(ms)	514	624	1372

33





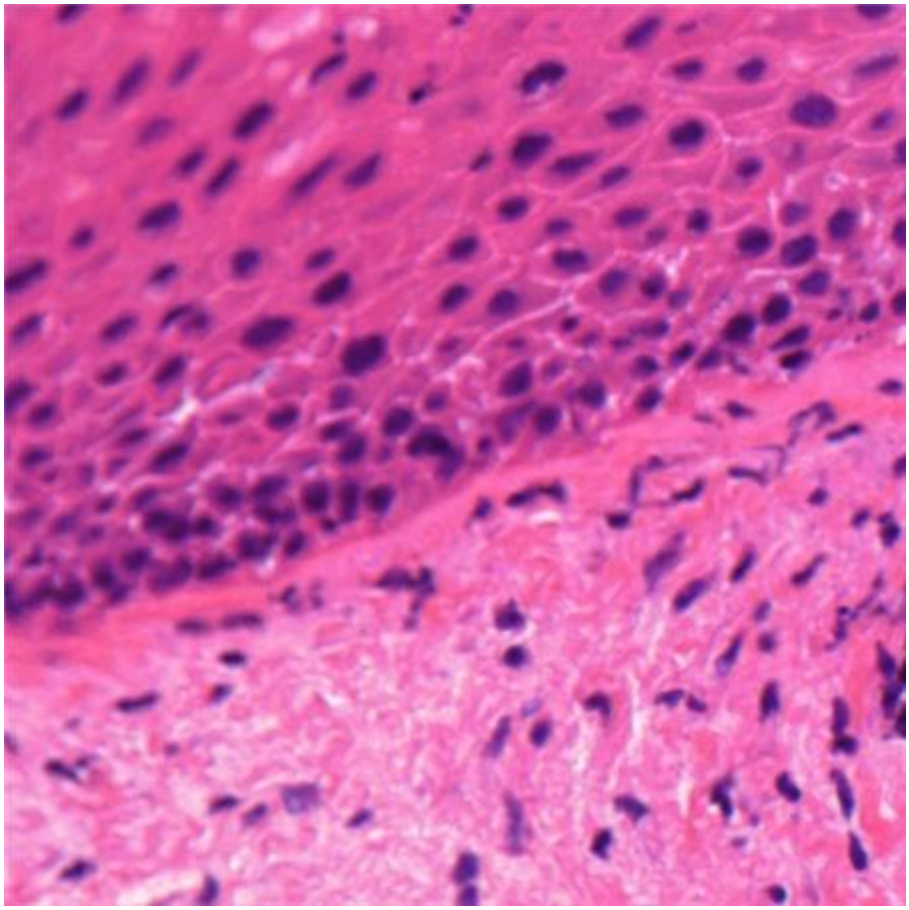
Wienert

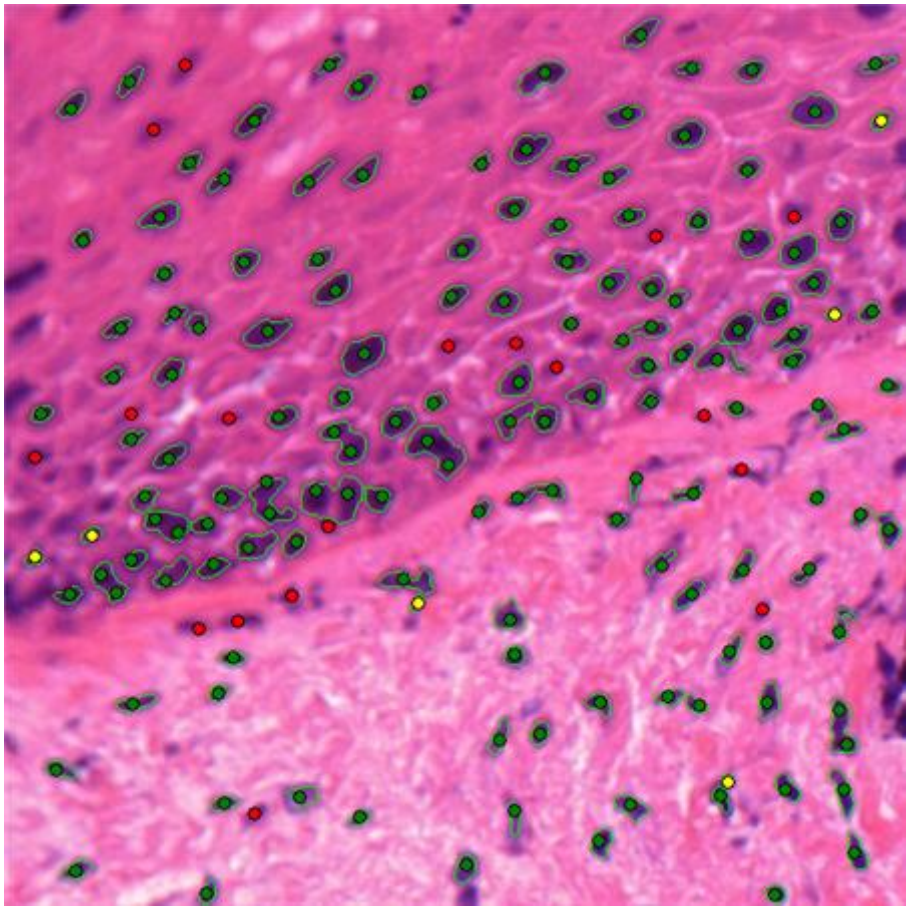


Al-Kofahi (optimized)

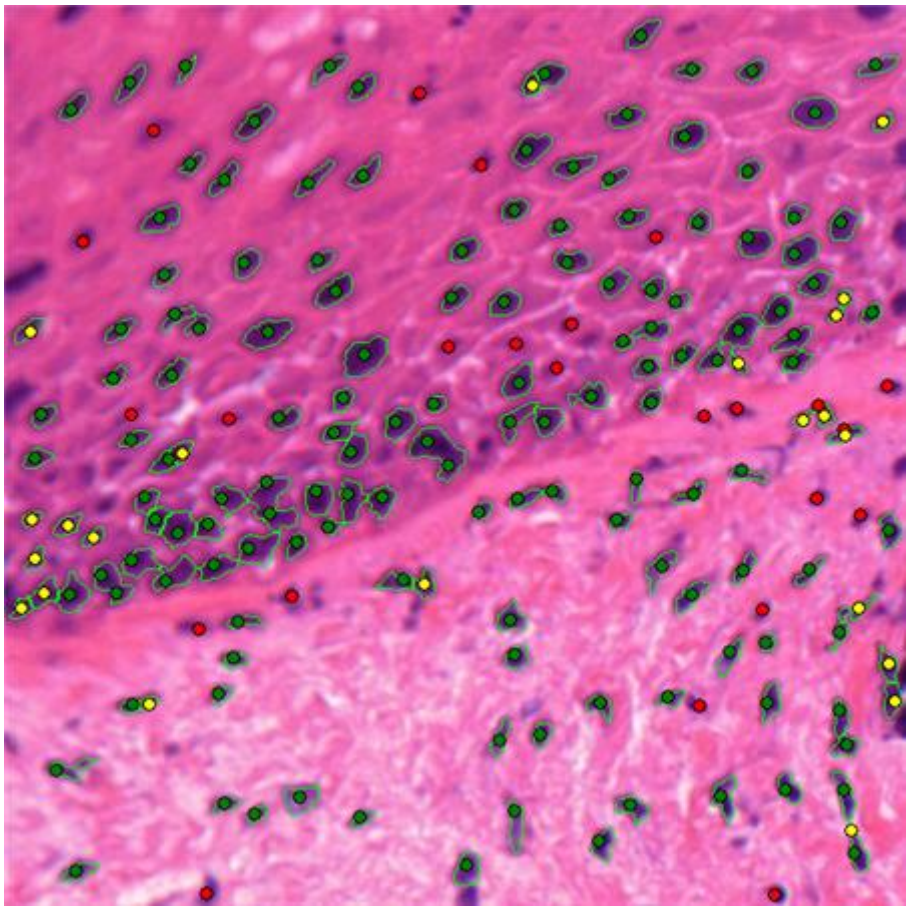
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	210	210	210
true positive count	177	191	190
false negative count	33	19	20
false positive count	21	174	74
precision	0,894	0,523	0,720
recall	0,843	0,910	0,905
conglomerate	0,904	0,906	0,926
time(ms)	717	639	2324

34





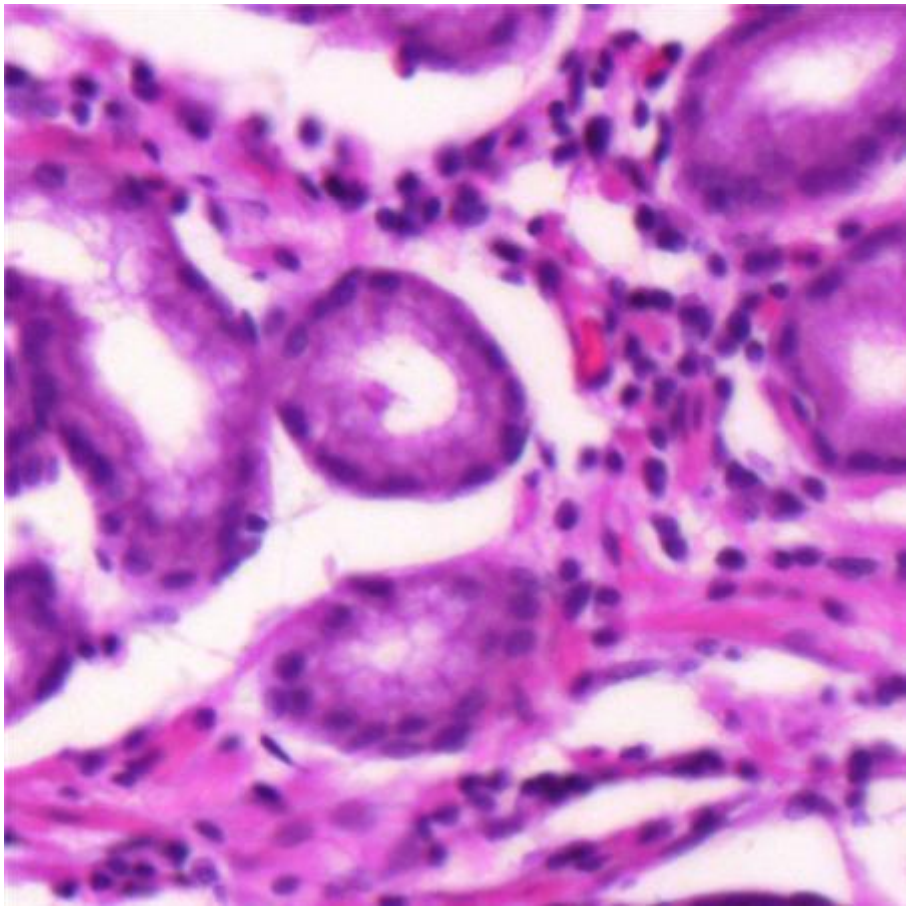
Wienert

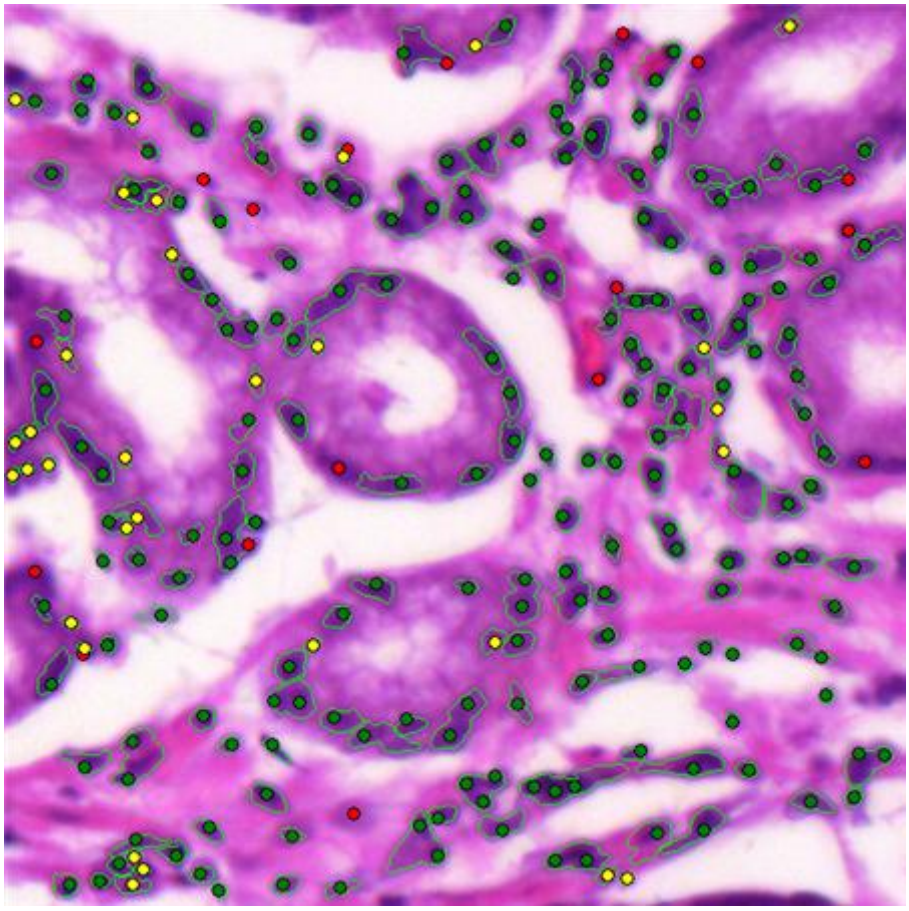


Al-Kofahi (optimized)

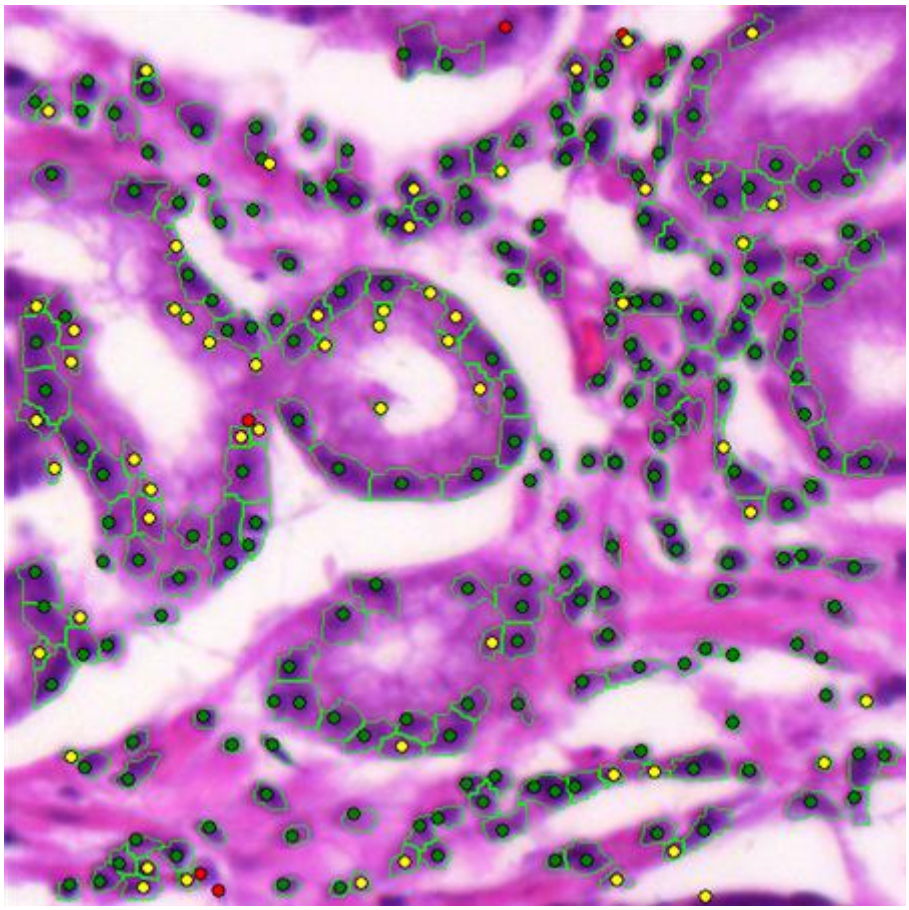
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	163	163	163
true positive count	145	139	139
false negative count	18	24	24
false positive count	6	34	22
precision	0,960	0,803	0,863
recall	0,890	0,853	0,853
conglomerate	0,945	0,986	0,986
time(ms)	499	795	1575

35





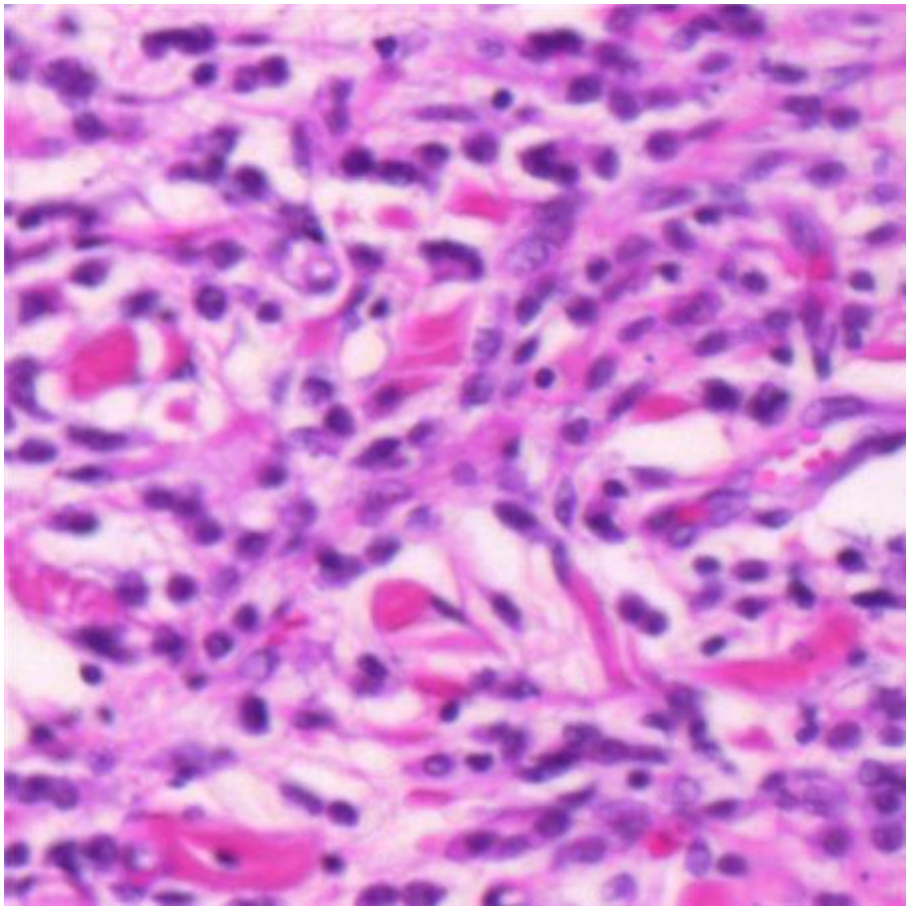
Wienert

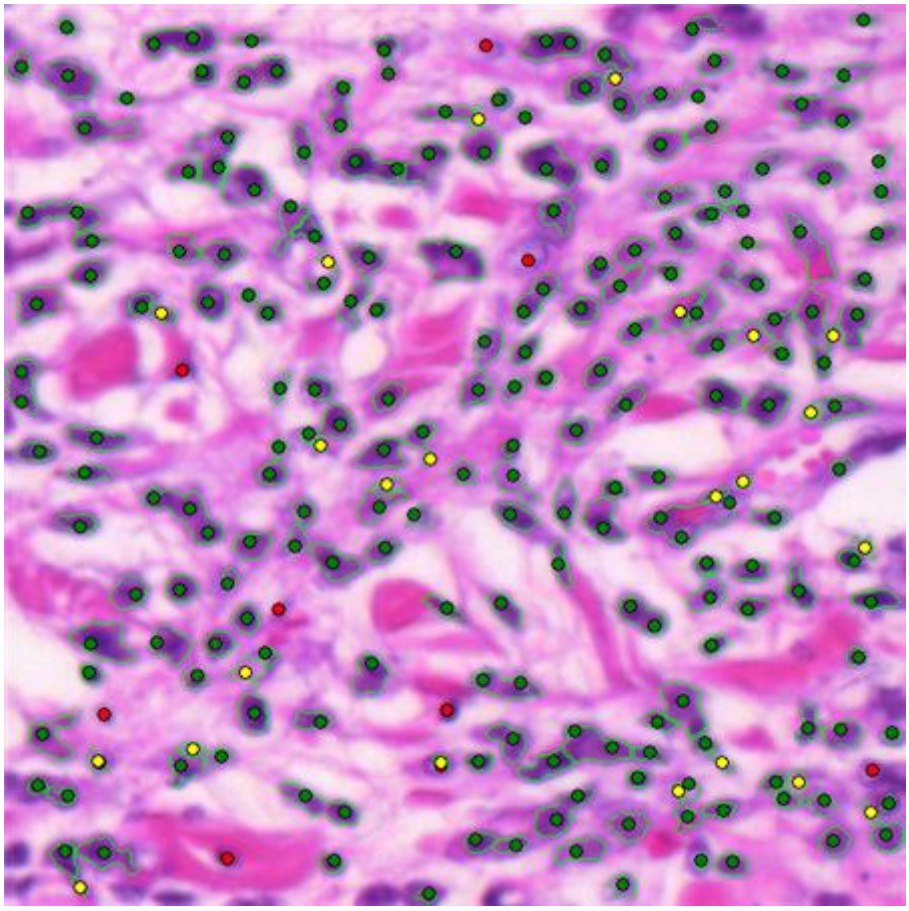


Al-Kofahi (optimized)

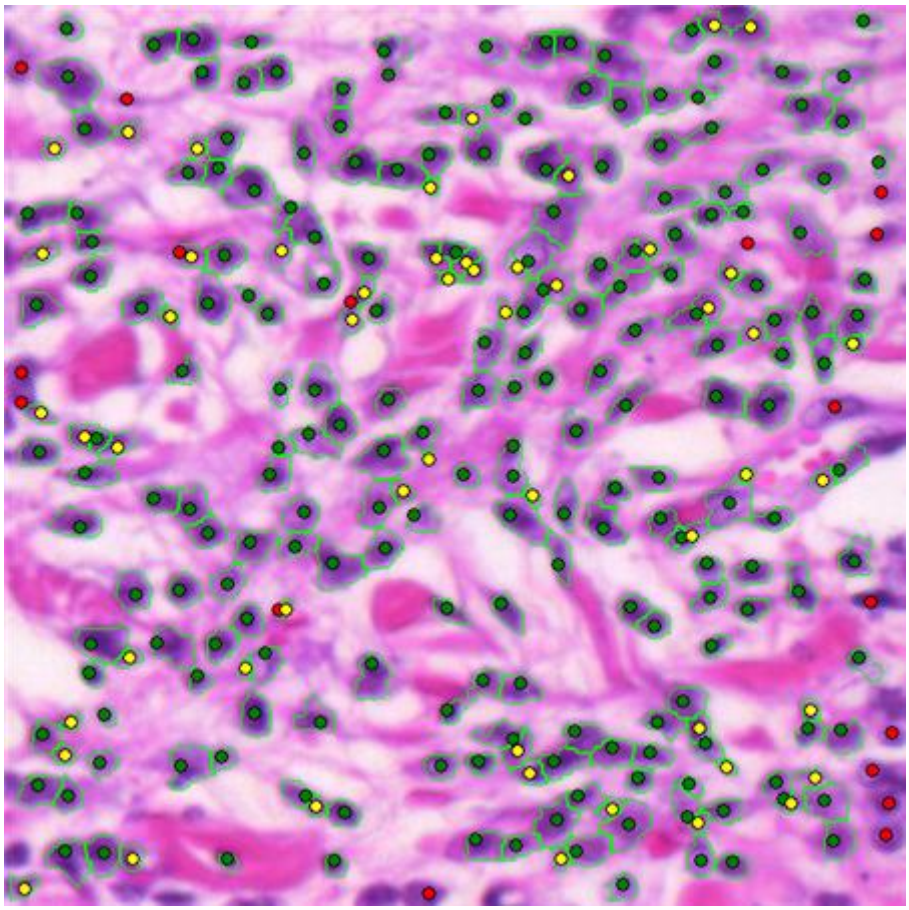
	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	211	211	211
true positive count	194	208	206
false negative count	17	3	5
false positive count	31	67	57
precision	0,862	0,756	0,783
recall	0,919	0,986	0,976
conglomerate	0,851	0,899	0,932
time(ms)	764	702	2262

36





Wienert



	Wienert	Al-Kofahi (automatic)	Al-Kofahi (optimized)
nuclei	223	223	223
true positive count	213	207	206
false negative count	10	16	17
false positive count	23	52	53
precision	0,903	0,799	0,795
recall	0,955	0,928	0,924
conglomerate	0,906	0,971	0,981
time(ms)	920	639	2121

Note S3: Example of influence of image focus on cell nucleus segmentation

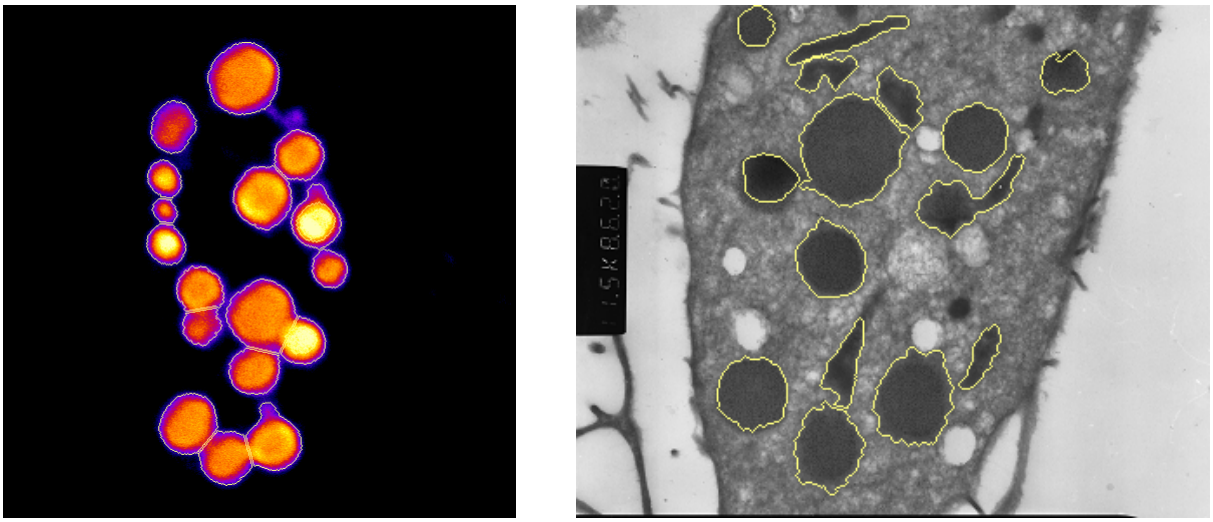
Cell nuclei that are out-of-focus typically have shallower gradients at the object border than in-focus nuclei. Because the proposed method selects for the locally most prominent contours and because no thresholding is used with respect to the object intensity or the gradient our method is relatively robust against image blur. To exemplify this we acquired a sequence of images with a microscope camera of the same cell but with different focus levels from sharp (Fig. 1a) to strongly blurred (Fig. 1e) in certain intervals using the microscope's fine drive. As can be seen in the below example figure (Fig. 1), while there is some small variation in the exact course of the contour, cell nuclei are detected robustly for all focus levels except the strongest out-of-focus example. However, this last test image hardly shows any discernible cell nucleus.

(a) (b) (c) (d) (e)

Fig. 1: Contour and cell nucleus detection performance as a function of decreasing focus levels ranging from sharp (a) to heavily blurred (e). Tile size 20µm x 20µm.

Note S4: Application in the field of correlative light electron microscopy

Correlative light electron microscopy (CLEM) [e.g. Vicidomini et al. 2008 and Vicidomini et al. 2010] uses a combination of different imaging modalities, such as confocal laser scanning microscopy (CLSM) and transmission electron microscopy (TEM) and relies on image segmentation and registration techniques to colocalize functional/molecular (CLSM) and ultrastructural (TEM) information. Because of its minimum-model approach our segmentation method may be applied to different imaging modalities such as CLSM and TEM (Fig. 1). It may therefore also be of use in future CLEM applications including automated image registration based on the segmentation results.



(a) (b)

Fig. 1: Rough Russel bodies segmented using the method presented in this paper of (a) confocal laser scanning microscopy (CLSM) and (b) transmission electron microscopy (TEM). Modifications of the algorithm: minContourLength=50, maxContourLength=500. Color deconvolution was not applied. Objects with a size below 350 pixels were removed. Additionally, (b) was reduced to 10% of the original size, smoothed with a 3x3 median filter and objects with a mean intensity outside the interval [25;85] were removed. Images were kindly provided by Giuseppe Vicidomini and Carlo Tacchetti (MicroSCoBiO Research Center, University of Genoa, Italy).

Methods S5: Seed point and extremal value detection

```
- input: I, grayscale image of size M x N
- input: maxLength, the maximum contour length in pixel

for v←0 to N-1
{
   /* Initialize the intensity variables */
   previousIntensity←I(0, v)
   intensity←I(1, v)

   /* Initialize the extremal variables */
   intensityOfMaxGradient←intensity
   minIntensity←previousIntensity
   maxIntensity←previousIntensity
   indexOfMinIntensity←0
   indexOfMaxIntensity←0
   indexOfMaxGradient←0
   maxGradient←-1

   /* “Walk” along the one dimensional image function */
   for u←1 to M-2
   {
      /* Read and store the value of the next pixel */
      nextIntensity←I(u+1, v)

      /* Update value and position of minimum intensity */
      if intensity<minIntensity
      {
         minIntensity←intensity
         indexOfMinIntensity←u
      }

      /* Update value and position of maximum intensity */
      if intensity>maxIntensity
      {
         maxIntensity←intensity
         indexOfMaxIntensity←u
      }

      /* Compute the local gradient */
      gradient←intensity-previousIntensity

      /* Turn into positive value if required */
      if gradient<0
      {
         gradient←-gradient
      }

      /* Update position and value of maximum local gradient */
      if gradient>maxLocalGradient
      {
         maxLocalGradient←gradient
         indexOfMaxLocalGradient←u
      }

      /* Determine if the slope is positive or negative */
      increasing←indexOfMinIntensity<indexOfMaxIntensity
      decreasing←indexOfMaxIntensity<indexOfMinIntensity

      /* Start 2D contour tracing if a local extremal value is reached */
      if (increasing && nextIntensity<intensity) || (decreasing && nextIntensity>intensity)
      {
         /* Compute the minimum and maximum intensity for the 2D contour tracing */
         if increasing
         {
            minObjectIntensity←intensityOfMaxGradient
            maxObjectIntensity←maxIntensity
         }
         else
         {
               maxObjectIntensity←intensityOfMaxGradient
               minObjectIntensity←minIntensity
         }
```

```
            /* Trace and store 2D contour starting where the local gradient is maximal Pixels
            /* with an intensity value between minIntensity and maxIntensity are interpreted as
            /* object pixel (foreground) */
            TraceAndStore2DContour(indexOfMaxLocalGradient, v, minIntensity, maxIntensity)

            /* Reset the extremal variables */
            minIntensity←intensity
            maxIntensity←intensity
            indexOfMinIntensity←u
            indexOfMaxIntensity←u
            maxGradient←-1
        }

        /* Shift the intensity values */
        previousIntensity←intensity
        intensity←nextIntensity
    }
}
```

Methods S6: Description of the "8M" algorithm [Hufnagl. et al.]

Object pixels are distinct from background pixels based on the specific object intensity range determined by scanning the image rows from left to right (see Methods section). Contour start pixels (seeds) are therefore at the objects left side. The initial search direction is "up". Based on the current contour pixel paraxial neighbours are tested clockwise. If one of these neighbours belongs to the current object the neighbour in counterclockwise direction is also tested. The contour continues with that pixel if it also belongs to the current object (see Fig. 1 below). The contour tracing stops when it reaches back to its seed and the contour pixel it would continue with is the same as the 2nd contour pixel. Alternatively the contour tracing is terminated if a maximum contour length (given by the total image perimeter by default) is exceeded.

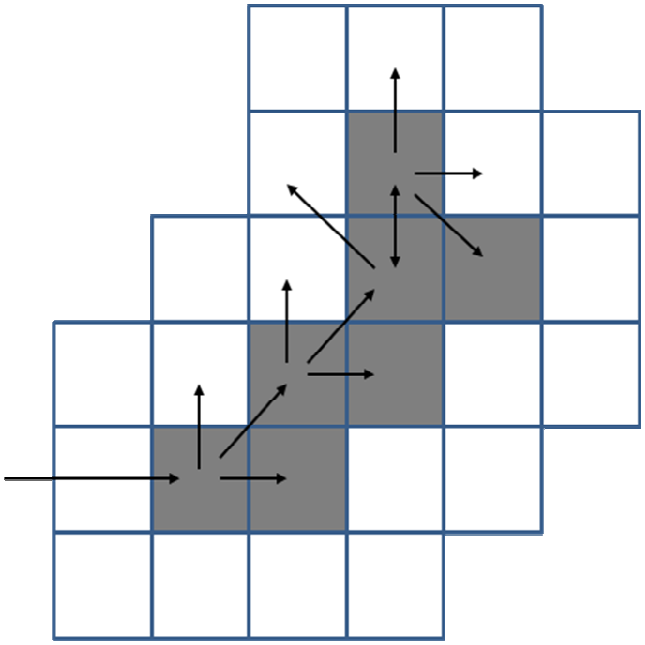


Fig. 1: Demonstration of contour tracing (Hufnagl. et al.).

Methods S7: Remove non-compact object pixels

```
- input: I, label image of size M x N
- input: distance, the initial distance value for testing compactness
- output: I₂, label image of size M x N

/* Initialization */
Set up a new label image I₂(u, v) of size M x N

/* Compute the Distance Transformation */
D←GetDistanceMap(I)

/* Decrement distance values that are not connected to a higher distance */
for d←distance-1 to 1
{
   for v←0 to N-1
   {
      for u←0 to M-1
      {
         /* Get the current distance */
         d0←D(u, v)

         /* No action when the current distance value is different */
         if d0<>d
         {
            continue
         }

         /* Get the distance values of the four neighboring pixels */
         d1←D(u+1, v)
         d2←D(u, v+1)
         d3←D(u-1, v)
         d4←D(u, v-1)

         /* Decrement the distance value when not connected to a distance value of d+1 */
         if d1≤d && d2≤d && d3≤d && d4≤d
         {
            D(u, v)←d-1
         }
      }
   }
}

/* Delete all labels that (now) are zero in the distance transformation */
for v←0 to N-1
{
   for u←0 to M-1
   {
      if D(u, v)>0
      {
         I₂(u, v)←I(u, v)
      }
      else
      {
         I₂(u, v)←0
      }
   }
}

return I₂
```

Methods S8: Separate concave objects

```
- input: I, label image of size M x N
- input: minDepth, the minimum depth of concave contour segments

do
{
   /* Process all contours in I */
   for each contour c contained in I
   {
      /* Find all concave points of c */
      candidates←DetectConcavePoints(c)

      /* Initialize the minimum score */
      minScore←positive infinity

      /* Select the cutting start point */
      for m←1 to count(candidates)-1
      {
         /* Process next candidate if depth is too low */
         if depth(candidates_m)<minDepth
         {
            continue
         }

         /* Select the cutting end point */
         for n←m+1 to count(candidates)
         {
            /* Process next candidate if depth is too low */
            if depth(candidates_n)<minDepth
            {
               continue
            }

            /* Compute the length and angles of the cutting line */
            r←GetDistance(candidates_m, candidates_n)
            a_m←GetAngle(candidates_m, candidates_n)
            a_n←GetAngle(candidates_n, candidates_m)

            /* Compute the angle between concave contour segment and cutting line */
            da_m←GetAngleBetween(candidates_m, a_m)
            da_n←GetAngleBetween(candidates_n, a_n)

            /* Compute the score of the cutting line */
            score←(r/(r+depth(candidates_m)+depth(candidates_m))+(da_m+da_n)/2π)/2

            /* Update the minimum score and the corresponding points */
            if score<minScore
            {
               minScore←score

               p_start←candidates_m
               p_end←candidates_n
            }
         }
      }

      /* Cut between the best two candidate if found */
      if minScore<positive infinity
      {
         CutLinear(I, p_start, p_end)
      }
   }
}
while cutting lines found
```