Evaluation of Binarization Algorithms

Bruce A. Bobier
Department of Computing and Information Science,
University of Guelph, Guelph, ON, N1G 2W1
Email:bbobier@uoguelph.ca

Michael Wirth
Department of Computing and Information Science,
University of Guelph, Guelph, ON, N1G 2W1
Email: mwirth@uoguelph.ca

***Abstract*—Twenty-one algorithms are compared for their performance at binarizing line drawings. And some other stuff happens.**

***Index Terms*—Binarization, Line Drawings, Performance Evaluation**

I. Purpose

Binarization, the process of labeling each pixel in a gray scale image as object data or background, is a common preprocessing step in most image analysis systems [1]. Numerous binarization algorithms have been presented in the literature and currently there are no known comparative evaluations of the algorithms' performance at binarizing line drawings. Several survey papers have however, evaluated such algorithms in the context of document image analysis and optical character recognition (OCR) [1–5].

Although the problem of recognizing characters in digitized textual documents is similar in principle to that of recognizing lines, arcs and shapes in line drawings, the intrinsic nature of each document class, and their subsequent image processing stages, suggest that techniques that are successful at binarizing textual documents are not necessarily amenable to processing line drawings. As such, this experiment attempts to conduct an objective and quantifiable evaluation of the suitability of a given binarization algorithm in the context of preprocessing architectural elevation drawings for vectorization. An architectural elevation drawing is a representation of a building's geometrical exterior as perceived from a horizontal viewpoint without dimensional perspective. The drawings are manually created with pen and paper and digitized as 8-bit gray scale images using a scanner or overhead camera. An immense number of architectural elevation drawings are contained in digital and analog archives worldwide, and in order for their contents to be fully realized for archival, retrieval and analytical tasks, a method of representing their semantic information in a non-visual format may be provided by preprocessing, vectorizing and classifying the drawings.

As there are no known comparative studies of algorithms for binarizing line drawings, current researchers and practitioners must select an algorithm based on the results of related studies (e.g. [1–3, 5]), prior experience, trial and error, or arbitrary selection. Thus, this experiment aims to solve the problem of selecting the binarization algorithm best suited to the preprocessing of line drawings for vectorization.

II. Method Details

In this experiment, 21 algorithms [6–27] are compared for their performance at binarizing line drawings. The central hypothesis to be tested by this experiment is that a statistically significant difference will be observed between the algorithms using the normalized cross-correlation approach that will allow a full ordering of algorithms to be established. One difficulty involved in comparing the algorithms on actual line drawings is that the ground truth, or optimal binary solution, is unknown. This has been overcome through the use of a data set of synthesized line drawings that characterize elevation drawings, and for which an optimal template is available. The diversity of test image size and complexity involved with this experiment should provide a wide spread of results which allows a significant difference between algorithms to be observed and quantified, if such a difference exists. Further, the evaluation of algorithms belonging to diverse algorithmic families should aid the verification of this hypothesis, as some algorithmic families may be better suited to binarizing line drawings than others.

If however, this hypothesis is not verified, an alternative hypothesis is that a single algorithm will perform uniformly better across the entire image set. This will also be verified or disproved by this experiment, because the experimental design permits the analysis of the performance and computation time of all algorithms on a per-image basis. However, by constructing the experiment using a repeated measures design, the statistical analysis of the results may also consider the performance and computation time of an algorithm across the entire data set, which supports the testing of the central hypothesis.

The conceptual definition of performance is defined as the degree to which image I resembles the corresponding template T. In operational terms, the performance of algorithm A for binarizing a gray scale image is defined as the value of the normalized cross-correlation between I and T. Normalized cross-correlation (NCC) is a statistical approach often used in template matching and pattern recognition problems that computes the probability of a binarized image being an instance of the template [28]. Formally, performance is defined by the normalized cross-correlation function for computing the degree of similarity between I and T:

$$\gamma(u,v) = \frac{\sum_{x,y}[I(x,y) - \overline{I}_{u,v}][T(x-u,y-v) - \overline{T}]}{\{\sum_{x,y}[I(x,y) - \overline{I}_{u,v}]^2 \sum_{x,y}[T(x-u,y-v) - \overline{T}]^2\}^{0.5}}$$

where $\overline{I}$ is the mean of I and $\overline{I}_{u,v}$ is the mean of $I(x,y)$ in the region of the template, and is calculated using the `normxcorr2(`I, T`)` function in the Image Processing Toolbox for Matlab [29]. If I matches T exactly, then γ will equal 1, whereas γ = 0 if I and T are entirely dissimilar.

Computation time is an additional factor that may influence an algorithm's suitability for binarizing a given type of line drawing. The operational definition of the computation time of algorithm A to binarize a gray scale test image is defined as the number of seconds required to binarize the image. The measurement of each algorithm's computation time is calculated using the built-in Matlab functions `tic` and `toc`, which return the number of seconds that have elapsed since `tic` was called.

To test the hypothesis that a statistically significant difference will be observed between the algorithms at binarizing a set of gray scale line drawings, the measurements of performance and computation time are subjected to post-hoc individual pairwise comparison using each algorithm's mean and a standard error value. These comparisons produce a p-value that is used to determine which of the algorithms are significantly different ($\alpha = 0.05$) and by how much.

A. Data Set

The data set used for this experiment consists of 120 images: 60 gray scale testing images and 60 template images used to quantify the performance of each algorithm. The data set is divided equally into three image subsets, with each subset having a scale of 256×256, 512×512 and 800×600. Each binarization algorithm processes all of the images in the data set, resulting in the generation of 2880 binary output images. The input, template and output images are non-interlaced `.png` files, as this format is bitmapped, uses lossless compression and preserves sharp edges. The file size of each template ranges from 1 to 4 Kb, while the test images range from 10 to 264 Kb, depending on the complexity of the background or degradation.

Each file is created in Adobe Photoshope CS2 [30] as a binary `.png` file composed of lines, arcs and simple shapes, and constitutes the template for later use during performance evaluation. Next, a copy of the template image is created, converted to 8-bit gray scale, and subjected to several forms of degradation or alteration to more accurately characterize an elevation drawing.

To degrade the foreground data in some images, uniform noise is added (with a parameter value between 15% and 45%) to mimic the dirt, aging and scanning artifacts that appear in elevation drawings. Frequently, elevation drawings that have been digitized with a scanner or overhead camera exhibit a small amount of blurring along the lines (∼2-4 pixels in width, depending on the resolution), due to ink bleeding in the paper and hardware limitations. This is simulated by applying a 2 pixel Gaussian blur to the lines in some of the test images using the Photoshop filter.

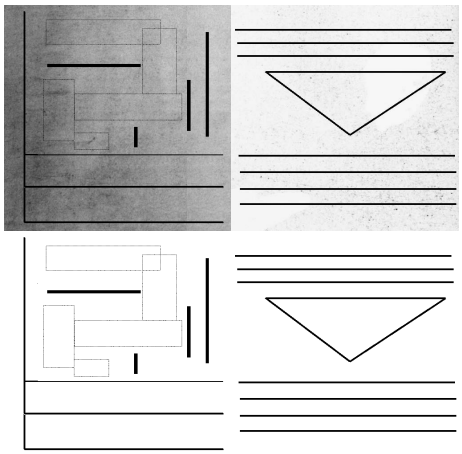

Fig. 1. Sample input images (left) and their templates (right)

To create the test images with altered backgrounds, each template's foreground data (black pixels) is placed in a separate layer, thereby allowing modifications to be made to the background data (white pixels). Several forms of background modification are also employed. In many images, background data is copied from actual elevation drawings in the Historic American Building Survey (HABS) [31], while in others, synthesized textures are created using Photoshop filters. In both cases, the new background data is placed beneath the template's foreground data, which remains untouched.

B. Algorithms

Twenty-one algorithms are evaluated in this experiment, each implemented in Matlab by Michael Wirth as part of the Microscopy and Imaging Toolbox [32]. The algorithms can be categorized into six groups, as outlined in the comprehensive literature survey by Sezgin and Sankur [5]:

1) Object attribute-based methods (AttribT), which search for a similarity measure between the gray level and binarized images (using for example, fuzzy shape similarity, edge coincidence, etc.);
2) Clustering-based methods (ClusterT), which cluster the gray level samples into two groups as either foreground or background data, or alternatively by modeling the image as a mixture of two Gaussian distributions;
3) Entropy-based methods (EntropyT), which use the difference of entropy between the foreground and background data or the cross-entropy between the original and binarized image;
4) Local methods (LocalT), which calculate an adaptive threshold value for each pixel based on the local image characteristics;
5) Histogram shape-based (ShapeT), which analyze the characteristics of the image's smoothed histogram, view-

ing it as a mixture of two Gaussian distributions attributed to foreground and background data; and

6) Spatial methods (SpatialT), which use higher-order probability distribution and/or correlation between pixels to establish a threshold value.

The twenty-one algorithms used in this experiment are listed in Table I, and further algorithmic details can be found in [32] and the associated reference. In addition to these, two of the algorithms, AttribT_Ramesh and AttribT_Yager, can be executed in variant forms, the former using either the sum of square errors or the sum of two variances, and the latter using either the Hamming or Euclidean metric for computing threshold values. Note that for evaluative and analytical purposes, each variant is treated as a separate algorithm. Only two other algorithms (LocalT_Bernsen and LocalT_Niblack) accept additional arguments beyond the image to be binarized. For Niblack's local algorithm, a 15×15 neighbourhood and adjustment parameter of $k = -0.2$ is selected for the experiment, following from [1, 32], while for Bernsen's local algorithm, values of 90, 16 and 1 are selected for the local contrast threshold, window size and high homogenous areas parameters respectively.

Algorithms		
AttribT_	BaradezMSA	[6]
	Huang	[10]
	moments	[20]
	Ramesh	[16]
	transition	[24]
	Yager	[21]
	Yumusak	[23]
ClusterT_	Brink	[8]
	Otsu	[15]
	Ridler	[25]
	Yanni	[22]
EntropyT_	Brink	[9]
	Kapur	[11]
	Li	[13]
	LiL	[12]
	Sahoo	[18]
	Shanbag	[19]
	Yen	[27]
LocalT_	Bernsen	[7]
	mAve	[26]
	Niblack	[14]
ShapeT_	unimodal	[17]

TABLE I
BINARIZATION ALGORITHMS USED FOR EVALUATION.

III. RESULTS

A total of ten runs are conducted on each of the 21 algorithms at binarizing the 60 image data set. As the algorithms are deterministic and the data set does not change, the NCC value of each algorithm-image pair is constant across each run. Thus, the NCC values are collected from a single run, while computation time data is collected from all ten runs. For each run, the algorithm name, iteration number, image number, NCC value, and computation time are collected.

Algorithm	Mean	Median	StdDev	Min	Max
attribT_BaradezMSA	0.569226	0.530652	0.202992	0.203241	0.996981
attribT_Huang	0.687451	0.932432	0.379734	0.002628	0.997165
attribT_moments	0.898139	0.918019	0.068305	0.729864	0.99131
attribT_Ramesh1	0.770029	0.789179	0.22225	0.00635	0.978208
attribT_Ramesh2	0.250925	0	0.39323	0	0.993745
attribT_Yager1	0.683956	0.930568	0.377973	0.002628	0.997165
attribT_Yager2	0.684309	0.930568	0.37814	0.002628	0.997165
clusterT_Brink	0.93649	0.94572	0.052502	0.762223	0.997165
clusterT_Otsu	0.936415	0.945599	0.052537	0.762223	0.997165
clusterT_Ridler	0.936484	0.94572	0.052516	0.762107	0.997165
clusterT_Yanni	0.811676	0.895292	0.246052	0	0.997165
entropyT_Brink	0.57682	0.561405	0.119266	0.320521	0.854516
entropyT_Kapur	0.614028	0.704027	0.253877	0.015692	0.943415
entropyT_Li	0.880164	0.939553	0.195514	0.154944	0.997165
entropyT_LiL	0.92949	0.941533	0.058827	0.717405	0.997165
entropyT_Shanbay	0.661826	0.660025	0.184887	0.017651	0.962726
entropyT_Yen	0.463138	0.46985	0.243909	0.007179	0.851005
localT_Bernsen	0.881277	0.907131	0.093592	0.545552	0.997165
localT_mAve	0.908303	0.922011	0.077843	0.654122	0.996849
localT_Niblack	0.487988	0.478945	0.086522	0.169757	0.676835
shapeT_unimodal	0.864657	0.930611	0.139252	0.39323	0.997165

TABLE II
SUMMARY OF NCC DATA FOR EACH ALGORITHM'S PERFORMANCE ON THE 60 IMAGES.

Algorithm	Mean	Median	StdDev	Min	Max
attribT_BaradezMSA	0.386537	0.398678	0.16696	0.022312	1.53228
attribT_Huang	0.127213	0.123961	0.047938	0.044865	0.436086
attribT_moments	0.040085	0.031251	0.097215	0.002581	1.53401
attribT_Ramesh1	0.007223	0.006976	0.003158	0.00597	0.049817
attribT_Ramesh2	3.26847	3.21496	2.08482	0.64457	7.23454
attribT_Yager1	0.056724	0.055313	0.018395	0.023805	0.090669
attribT_Yager2	0.055498	0.054064	1.82E-02	0.023439	0.098499
clusterT_Brink	0.032262	0.019785	0.030579	0.017644	0.216629
clusterT_Otsu	0.004903	4.80E-03	0.001294	0.003825	0.030006
clusterT_Ridler	0.006109	0.005665	0.00438	0.001745	0.040934
clusterT_Yanni	0.001505	0.001407	8.33E-04	6.74E-04	0.009538
entropyT_Brink	0.036491	0.036452	0.001391	0.033686	0.054843
entropyT_Kapur	0.293961	0.27528	0.111458	0.099838	0.480385
entropyT_Li	0.075896	0.062869	0.040031	0.00461	0.196157
entropyT_LiL	0.660154	0.661647	0.02017	0.613333	0.882634
entropyT_Shanbay	0.032258	0.032618	0.003743	0.011959	0.07089
entropyT_Yen	0.003618	0.003318	0.002319	0.002166	0.046796
localT_Bernsen	0.091852	0.092106	0.02655	0.009117	0.148374
localT_mAve	0.127845	0.119715	0.084713	0.026979	0.291845
localT_Niblack	18.5466	18.1334	12.1792	3.75117	39.3944
shapeT_unimodal	0.001815	0.00109	0.013568	5.16E-04	0.330134

TABLE III
SUMMARY OF COMPUTATION TIME DATA FOR EACH ALGORITHM'S PERFORMANCE ON THE 60 IMAGES.

In terms of NCC values (Table II), the highest means are generated by clusterT_Brink [8], clusterT_Ridler [25] and clusterT_Otsu [15] respectively, while the lowest mean values are generated by attribT_Ramesh2 [16], entropyT_Yen [27] and localT_Niblack [14]. In terms of computation time (Table III), the best (lowest) means are produced by clusterT_Yanni [22], shapeT_unimodal [17] and entropyT_Yen, while the three highest mean times are produced by localT_Niblack, attribT_Ramesh2 and entropyT_LiL [12] respectively.

IV. ANALYSIS AND DISCUSSION

To perform statistical analysis of the experimental data, regression analysis is first conducted on a normal probability plot to determine if the data are normally distributed. For the NCC and computation time data, r^2 values of 81.2% and 24% are computed respectively, both of which fall below the

confidence level of 95%. Because of the non-normality of the data, non-parametric statistics are employed on the ranked data for the remaining statistical calculations.

For both NCC and computation time, an ANOVA ($\alpha = 0.05$) is calculated for all algorithms, which produces p-values of 1.42E-123 and 0 respectively, and indicates that there is a statistically significant difference between groups. Data Desk [33] is then used to conduct a post-hoc individual pairwise comparison of algorithms to test the central hypothesis. The post-hoc comparison utilizes the difference of each pair of algorithms' means and a standard error to compute a p-value, such that a p-value that is less than $\alpha = 0.05$ reflects a statistically significant difference between the paired algorithms. A summary of these post-hoc comparisons based on NCC values is given in Table IV, where the algorithm name in cell (i, j) has the statistically significantly higher mean NCC of the algorithms in the i^{th} row of the leftmost column and the j^{th} column of that section's top row. A blank cell indicates that there is not a statistically significant difference between the two algorithms. Additionally, a graph-based summary of the post-hoc comparisons is given in Fig. 2 as a partial ordering of algorithms, with the better performing algorithms situated atop the worse performing algorithms. Each node indicates that the contained algorithms perform better than those contained in its child nodes, as well as all nodes below its child. Further, there is not a statistically significant difference between algorithms in the same node or nodes of equal depth in the graph.

Given that the graph is not a linear structure and a full ordering cannot be established, the hypothesis that a statistically significant difference exists between algorithms based on their NCC that allows for a full ordering, is rejected. The hypothesis that a single algorithm performs better across the entire image set is also rejected, because although clusterT_Brink obtains the highest NCC value on every image in the data set, these values are not statistically significantly better than the other algorithms that achieved the highest NCC value on some but not all images.

Post-hoc comparisons of the computation time of each pair of algorithms are also conducted, the results of which are summarized as a partial ordering graph (Fig. 3). Again, a full ordering of algorithms cannot be established based on computation time for binarizing the test images, as there is not a statistically significant difference between the 18 fastest algorithms. The hypothesis that no single algorithm performs better across the entire image set in terms of computation time is also rejected by this experiment, as the lowest computation time varies between clusterT_Yanni, entropyT_Yen and shapeT_unimodal, despite the latter providing the fastest time on 98.83% of the tests.

Cross-correlation between algorithms and images to determine on which images each algorithm struggles cannot be computed, as there is an insufficient number of instances of each variable for statistical analysis. However, analysis of NCC values for all algorithm-image pairs shows a mean of 0.735, median of 0.861, minimum of 0, maximum of 0.997, and standard deviation of 0.286. Similarly, statistical analysis

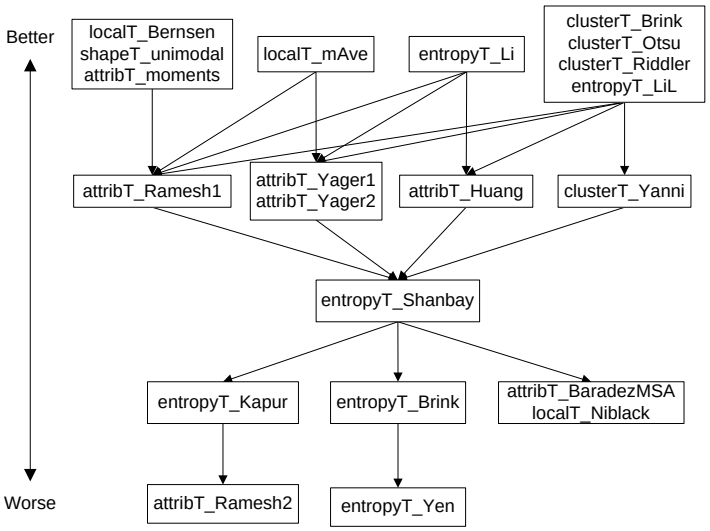

Fig. 2. Partial ordering of algorithms from post-hoc comparisons of NCC.

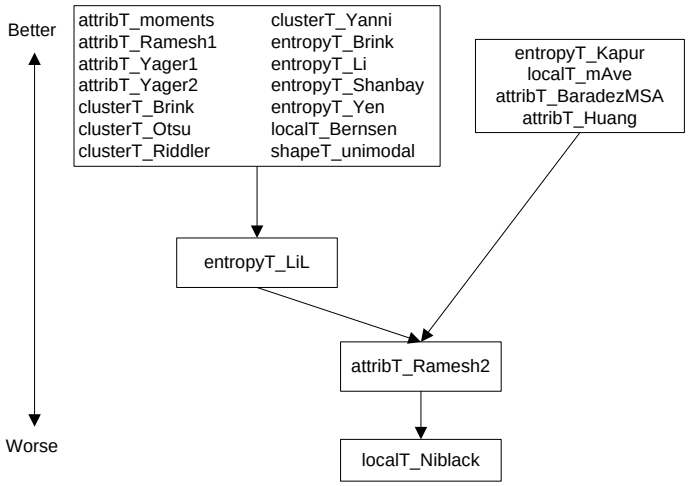

Fig. 3. Partial ordering of algorithms from post-hoc comparisons of computation time.

of the interaction between each algorithm's performance cannot be conducted due to insufficient data, although post-hoc analysis of all NCC and computation time on each image size is performed. Analysis of the NCC values (Table V) shows a statistically significant difference ($\alpha = 0.05$) between the mean values of the 20 256×256 and 512×512 images ($p = 0.000026782$), as well as between the 256×256 and 800×600 images ($p = 0.00266498$). Analysis of the computation time values for each image size (Table VI) shows a statistically significant difference of computation time between the 20 256×256 and 512×512 images ($p = 2.00E - 13$) only.

From the analysis of these results, several unexpected results are noted. First, the performance of localT_Niblack is among the lowest, which is surprising given that the algorithm has been shown to provide superior performance in comparative

	attribT_ BaradezMSA	attribT_ Huang	attribT_ moments	attribT_ Ramesh1	attribT_ Ramesh2	attribT_ Yager1	attribT_ Yager2
shapeT_unimodal	unimodal			unimodal	unimodal		
localT_Niblack		Huang	moments	Ramesh1		Yager1	Yager2
localT_mAve	mAve			mAve	mAve	mAve	mAve
localT_Bernsen	Bernsen			Bernsen	Bernsen		
entropyT_Yen		Huang	moments	Ramesh1		Yager1	Yager2
entropyT_Shanbay		Huang	moments	Ramesh1	Shanbay	Yager1	Yager2
entropyT_LiL	LiL	LiL		LiL	LiL	LiL	LiL
entropyT_Li	Li	Li		Li	Li	Li	Li
entropyT_Kapur		Huang	moments	Ramesh1		Yager1	Yager2
entropyT_Brink		Huang	moments	Ramesh1		Yager1	Yager2
clusterT_Yanni	Yanni				Yanni		
clusterT_Ridler	Ridler	Ridler		Ridler	Ridler	Ridler	Ridler
clusterT_Otsu	Otsu	Otsu		Otsu	Otsu	Otsu	Otsu
clusterT_Brink	Brink	Brink		Brink	Brink	Brink	Brink
attribT_Yager2	Yager2				Yager2		/
attribT_Yager1	Yager1				Yager1	/	
attribT_Ramesh2		Huang	moments	Ramesh1	/		
attribT_Ramesh1	Ramesh1		moments	/			
attribT_moments	moments		/				
attribT_Huang	moments	/					
attribT_BaradezMSA	/						

	clusterT_ Brink	clusterT_ Otsu	clusterT_ Ridler	clusterT_ Yanni	entropyT_ Brink	entropyT_ Kapur	entropyT_ Li
shapeT_unimodal					unimodal	unimodal	
localT_Niblack	Brink	Otsu	Ridler	Yanni			Li
localT_mAve					mAve	mAve	
localT_Bernsen					Bernsen	Bernsen	
entropyT_Yen	Brink	Otsu	Ridler	Yanni			Li
entropyT_Shanbay	Brink	Otsu	Ridler	Yanni			Li
entropyT_LiL				LiL	LiL	LiL	
entropyT_Li					Li	Li	/
entropyT_Kapur	Brink	Otsu	Ridler	Yanni		/	
entropyT_Brink	Brink	Otsu	Ridler	Yanni	/		
clusterT_Yanni	Brink	Otsu	Ridler	/			
clusterT_Ridler			/				
clusterT_Otsu		/					
clusterT_Brink	/						

	entropyT_ LiL	entropyT_ Shanbay	entropyT_ Yen	localT_ Bernsen	localT_ mAve	localT_ Niblack	shapeT_ unimodal
shapeT_unimodal		unimodal	unimodal			unimodal	/
localT_Niblack	LiL	Shanbay		Bernsen	mAve	/	
localT_mAve		mAve	mAve		/		
localT_Bernsen		Bernsen	Bernsen	/			
entropyT_Yen	LiL	Shanbay	/				
entropyT_Shanbay	LiL	/					
entropyT_LiL	/						

TABLE IV
SUMMARY OF POST-HOC COMPARISONS BASED ON NCC VALUES. THE ALGORITHM IN CELL (i, j) HAS THE STATISTICALLY SIGNIFICANTLY HIGHER MEAN NCC OF THE ALGORITHMS IN THE i^{th} ROW OF THE LEFTMOST COLUMN AND THE j^{th} COLUMN OF THAT SECTION'S TOP ROW.

SIZE	Mean	Median	Min	Max	StdDev
256x256	0.698569	0.835453	0	0.993745	0.299481
512x512	0.748921	0.871161	0	0.993745	0.281592
800x600	0.757666	0.875383	0	0.996981	0.273687

TABLE V
SUMMARY OF POST-HOC ANALYSIS OF NCC ON EACH IMAGE SIZE.

Scale	Mean	Median	Min	Max	StdDev
256x256	0.319654	0.035156	0.000516	5.632906	0.88898
512x512	1.095231	0.044479	0.001048	19.52365	3.764706
800x600	1.99351	0.042607	0.001638	39.39436	7.235143

TABLE VI
SUMMARY OF POST-HOC ANALYSIS OF COMPUTATION TIME ON EACH IMAGE SIZE.

evaluations of algorithms as a preprocessing step for OCR [1]. Further, the computation time of localT_Niblack is statistically significantly worse than all other evaluated algorithms, which is also surprising when considering its frequency of use in the OCR domain [4, 5].

V. CONCLUSION

This experiment evaluates 21 binarization algorithms to determine which algorithm is best suited for the preprocessing of elevation drawings for vectorization and reports that a statistically significant full ordering of algorithms based on their NCC performance cannot be established. Sixty 8-bit gray scale images of dimensions 256×256, 512×512 or 800×600 are binarized ten times by each algorithm, with computation time (seconds) and performance (the normalized cross-correlation value between the binary output image and its corresponding template) being recorded for each run. Post-hoc individual pairwise comparisons are made, between both the computation time and NCC of each algorithm, and used to establish a partial ordering of algorithms for each variable. As a full ordering cannot be established from the test data values, no single

algorithm can be seen as a panacea for binarizing elevation drawings. However, generally Brink's [8], Otsu's [15] and Ridler and Calvard's [25] clustering-based methods are noted as being suitable candidates for this problem, as they provide quick, accurate results for the data set.

Several new hypotheses are noted from this experiment, the first being that a statistically significant full ordering of this experiment's nine best performing algorithms (as they cannot be statistically discriminated) can be established based on their NCC values from binarizing a larger, different data set. Similarly, it is hypothesized that a full ordering of these algorithms can be established by using an alternative similarity metric, such as the Hausdorff distance [34, 35], for comparing each algorithm's output with the image's corresponding template. A third hypothesis is that a full ordering of each algorithm's computation time can be established by using a larger data set comprised of images with larger dimensions (i.e. $> 1500 \times 1500$).

Future work may include utilizing this experimental framework in the evaluation of text/graphics segmentation algorithms, which typically follow binarization in the preprocessing of elevation drawings for vectorization. Similarly, vectorization and direct-recognition algorithms may also be compared using this framework, with rasterized CAD-based line drawings serving as templates, and their degraded/modified versions serving as input images. Finally, the results of this experiment may also be applied in several other domains, including for example, the binarization of modern maps for GIS input, historic maps and diagrams for indexing and analysis, as well as the binarization of figures and images in primarily textual documents.

REFERENCES

[1] Ovind Due Trier and Anil K. Jain. Goal-directed evaluation of binarization methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1191–1202, 1995.

[2] C.A. Glasbey. An analysis of histogram-based thresholding algorithms. *Graphical models and image processing*, 55(6):532–537, November 1993.

[3] S.U. Lee, S.Y. Chung, and R.H. Park. A comparative performance study of several global thresholding techniques for segmentation. *Computer Vision and Graphic Image Processing*, 52(2):171–190, November 1990.

[4] P.W. Palumbo, P. Swaminathan, and S.N. Srihari. Document image binarization: Evaluation of algorithms. *SPIE Applications of Digital Image Processing*, 697:278–285, 1986.

[5] Mehmet Sezgin and Bülent Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–165, January 2004.

[6] M.O. Baradez, C.P. McGuckin, N. Forraz, R. Pettengell, and A. Hoppe. Robust and automated unimodal histogram thresholding and potential applications. *Pattern Recognition*, 37(6):1131–1148, June 2004.

[7] J Bernsen. Dynamic thresholding of grey-level images. In *International Conference on Pattern Recognition*, pages 1251–1255, 1986.

[8] A.D. Brink. Gray-level thresholding of images using a correlation criterion. *Pattern Recognition Letters*, 9(5):335–341, June 1989.

[9] A.D. Brink and N.E. Pendock. Minimum cross-entropy threshold selection. *Pattern Recognition*, 29(1):179–188, January 1996.

[10] L.K. Huang and M.J.J. Wang. Image thresholding by minimizing the measures of fuzziness. *Pattern Recognition*, 28(1):41–51, January 1995.

[11] J.N. Kapur, P.K. Sahoo, and A.K.C. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Graphical Models and Image Processing*, 29(3):273–285, March 1985.

[12] Chun Hung Li and C. K. Lee. Minimum cross entropy thresholding. *Pattern Recognition*, 26(4):617–625, 1993.

[13] C.H. Li and P.K.S. Tam. An iterative algorithm for minimum cross-entropy thresholding. *Pattern Recognition Letters*, 19(8):771–776, June 1998.

[14] Wayne Niblack. *An Introduction to Digital Image Processing*. Prentice Hall, New Jersey, 1986.

[15] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[16] N. Ramesh, J.H. Yoo, and I.K. Sethi. Thresholding based on histogram approximation. *IEEE Proceedings Vision, Image and Signal Processing*, 142(5):271–279, October 1995.

[17] P.L. Rosin. Unimodal thresholding. *Pattern Recognition*, 34(11):2083–2096, November 2001.

[18] Wilkins Sahoo, P., C., and J. Yeager. Threshold selection using Renyi's entropy,. *Pattern Recognition*, 30:71–84, 1997.

[19] Abhijit G. Shanbhag. Utilization of information measure as a means of image thresholding. *CVGIP: Graphical Model and Image Processing*, 56(5):414–419, 1994.

[20] H Tsai. Moments preserving thresholding: A new approach. *Computer Vision, Graphics and Image Processing*, 29(3):377–393, 1985.

[21] R Yager. On the measures of fuzziness and negation. *International Journal of General Systems*, 5:221, 1979.

[22] M.K. Yanni and E. Horne. A new approach to dynamic thresholding. In *EUSIPCO European Signal Processing Conference*, volume 1, pages 34–44, 1994.

[23] N. Yumusak, F. Temurtas, O. Cerezci, and S. Pazar. Image thresholding using measures of fuzziness. In *24th Annual Conference of the IEEE Industrial Electronic Society*, pages 1300–1305, 1998.

[24] Y. J. Zhang and J. J. Gerbrands. Transition region determination based thresholding. *Pattern Recognition Letters*, 12(1):13–23, 1991.

[25] T.W. Ridler and S. Calvard. Picture thresholding using an iterative selection method. *System, Man and Cybernetics*, 8(8):629–632, August 1978.

[26] P. Wellner. Adaptive thresholding on the DigitalDesk. Technical report: EPC-93-110, EuroPARC, 1993.

[27] J.C. Yen, F.J. Chang, and S. Chang. A new criterion for automatic multilevel thresholding. *IEEE Transactions in Image Processing*, 4(3):370–378, March 1995.

[28] Lisa Gottesfeld Brown. A survey of image registration techniques. *ACM Computing Surveys*, 24(4):325–376, 1992.

[29] Mathworks Inc. Matlab - image processing toolbox. http://www.mathworks.com/ access/helpdesk/help/toolbox/images/, 2006.

[30] Adobe. Photoshop CS2. www.adobe.com/products/photoshop/, 2007.

[31] Historic American Buildings Survey/Historic American Engineering Record (HABS/HAER). http://memory.loc.gov/ammem/collections/ habs_haer, 2007.

[32] Purdue University Cytometry Laboratories. Microscopy & Imaging 3. http://www.cyto.purdue.edu, August 2005.

[33] P.F. Velleman. Data desk 4.2: Data description. Ithaca, N.Y., 2006.

[34] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[35] C. Robertson and J.A. Robinson. Page similarity and the Hausdorff distance. In *Seventh International Conference on Image Processing and Its Applications*, volume 2, pages 755–759, 1999.