Sketchs characterization and compression for nomadic computing

Nelson Baloian, Ramón Cruzat, Richard Ibarra
Departamento de Ciencias de la Computación
Universidad de Chile
{nbaloian,rcruzat,ribarra}@dcc.uchile.cl
Blanco Encalada 2120, Santiago de Chile.

Javier Bustos-Jiménez
ORAND
Estado 360 of 702A, Santiago, Chile
javier.bustos@orand.cl
Estado 360, Piso 7, Santiago de Chile.

Abstract— **Nomadic computing often refers to people using computer support working anywhere, anytime and not necessarily attached to a specific location or time of the day. Mobile computing and wireless ad-hoc networks are important elements in this kind of scenarios. Nowadays it is common to see architects, engineers, geologists and/or designers working on the field and sharing ideas on a collaborative media using sketches and freehand writing. On this scenario, data overflow among the peer applications often happens, since the sketches have to be distributed among all participants in real time. This paper presents a work on characterization and compression algorithms for sketches, with and without loss of information, to be used on mobile devices, in order to reduce the data traffic. We focused our study on three parameters of the algorithms: time required characterizing and compressing the sketch, size of the resulting information and human perceived of lost information. Our results present algorithms with a compression ratio lower than 1% the size of the original image without information loosing under human perception.**

Nomadic computing, mobile computing, peer-to-peer collaborative systems

I. INTRODUCTION

In the last decade, mobile collaborative applications have attracted the interest of many researchers and nowadays we can find a vast number of works reporting on this kind of applications in various spheres in the literature. For instance, using mobile devices for adding new information to the environment (known as "*Augmented Reality*" [1], [2]), using mobile information for commerce [3], or education [4], [5].

An important factor that contributed to this area has certainly been the rapid development experimented by mobile hardware devices as well as networks, reaching almost any place of a developed city, with the consequent fall in the acquisition costs for mobile devices as well as connectivity services.

A very important consequence of this development has also been that the traditional working style of people depending on computing resources to do their work has dramatically changed. Today it is common to find people working anywhere, anytime not necessarily attached to a specific location or time of the day [6]. This working style has been named as nomadic computing by some authors [7].

The kind of work people do outside offices is often characterized by activities involving on-site collaborative design sketching [8]. Some examples are:

- Architects working jointly on a construction site using sketches to exchange ideas about re-designing facilities [9].
- Engineers conducting an on-site inspection, trying to find possible deficiencies and improvements of the facilities, exchanging graphic sketches while moving around the premises [10].
- Geologists and/or topographers on a field trip jointly drawing a map of the physical characteristics of a geographical area [11].
- People trying to take a decision based on ideas generated on the fly [12].
- Computer supported pervasive learning through the use of patterns and sketches [13].

All these activities involve data collecting on the field and creating initial sketches which will be later refined in the office with the help of more powerful desktop computers working on the data initially collected with handheld computer devices wirelessly interconnected [14].

A handheld's most natural data-entry mode is using the stylus (a.k.a. a pen-based or freehand-input-based system). This enables users to easily write down their ideas and/or draw design sketches imitating the use of pen and paper [15]–[17].

However, there are some problems when using sketches in collaborative working scenarios with mobile devices. In fact, synchronizing a workspace shared by many applications in which the shared data are sketches generates a high rate of data traffic inside the network, which in most cases is a mobile ad- hoc network (MANET).

On the other side, mobile devices have limited memory/processor capability and the MANETs they are able to build have low band- width and latency. Therefore, it seems a good idea to put some effort in order to reduce the quantity of data shared, without losing the shared information significantly.

In this article we study how to perform a characterization and compression of sketches with minimal information loose in order to reduce the data traffic in mobile collaborative applications using sketches as the main shared information among participants in a working group. By characterization of a sketch we mean the representation of the same sketch using fewer data than the original set of points generated by the sketching application. By information loose we mean the human perceived difference between the original sketch and the characterized one.

The rest of the article is organized as follows: the following section present the problem we are trying to solve in detail, in Section III we present the algorithms used on this study in order to characterize and compress sketches, in Section IV we present our Java Tool for the study of sketches characterization algorithms called X, followed by the study itself in Section V and main conclusions on Section VI.

II. The problem

Mobile computing devices which can provide the needed computing support for the development of the scenarios described in the previous section have still some problems which are not present on desktop computers:

a) Communication via radio-networks implies low-bandwidth and high latency and high packet loss rate.
b) Devices have limited memory and CPU.

Point a) implies that there will be necessary problems when transmitting large quantities of data among the mobile devices. The scenario of peer-to-peer applications sharing handwriting and sketches is very intensive in data sharing since sketches are normally characterized as a collection of strokes, which in turn are characterized by a very large set of points. Moreover, every sketch produced by one of the members must be transmitted to all peers taking part of the working session. Point b) implies that a mobile device will have troubles performing sophisticated compression and/or characterization algorithms with low ratio of information loosing.

Our study aims to find a suitable algorithm to perform a small, quick and accurate characterization and compression procedure of sketches for its use on mobile environments. To perform such study, we developed a Java application called "X" (details in Section 3). This application allows a user to draw a sketch and perform various characterization and/or compression algorithms over it, which have previously codified and included as a plug-in of "X" following its definition in an interface file. For each of the performed algorithms "X" calculates the following values:

- Time used for the characterization and compression of the sketch (***t***).
- Size of the resulting characterization (***s***).
- Ratio (%) of information loosing due the compression (***l***). This is calculated as $\|O - R\|/\|O\|$, where O is the pixel matrix of the original image and R is the pixel matrix of the reconstructed image using the characterization and/or compression algorithm being tested.

Next Section presents the studied algorithms, followed by the main results of our metrics and conclusions of our study.

III. Algorithms for characterization of sketches

We studied four algorithms which characterize and compress sketch information consisting of a set of points. These are based on different attributes. The first one is based on the distance the points of a certain stroke are separated from each other, the second is based on the area the points are located, the fourth one scales down the original whole original pictures and the fourth makes a "zipping" of the sketch information.

A. Based on Distance

As it is presented on Figure 1, this algorithm is quite simple but powerful: it removes points on the sketch based on the distance between two neighboring points. The idea behind this algorithm is that if two points are close enough (distance less than a parameter e), it means one of them does not provides useful information for the whole sketch and it could be deleted.

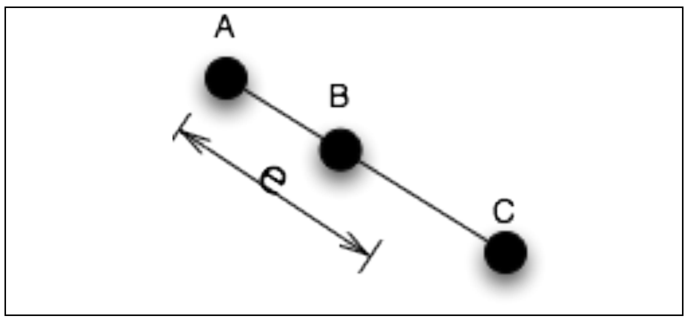


Figure 1. Distance-based algorithm. If distance AB is less than e point B can be discarded

In the Figure 1, notice that the distance between points A and B is lower than ***e***, and the distance between A and C is greater than ***e***. Therefore the point B could be deleted.

B. Based on Area

Similar to the previous idea, this algorithm uses the difference of the area of two polygons formed by three neighboring points ABC and an additional one X in the following way: Consider X to be the vertex of a right-angled triangle where A and C are the starting and ending points of the hypotenuse (see Figure 2). If the difference of the areas of the trapezoid defined by the points ABCX and triangle

ACX is small enough (say |area(ABCX)-area(ACX)| < ***e***), it means that the points A, B and C are almost on a same line and point B does not provide useful information for the whole sketch, so it can be deleted.

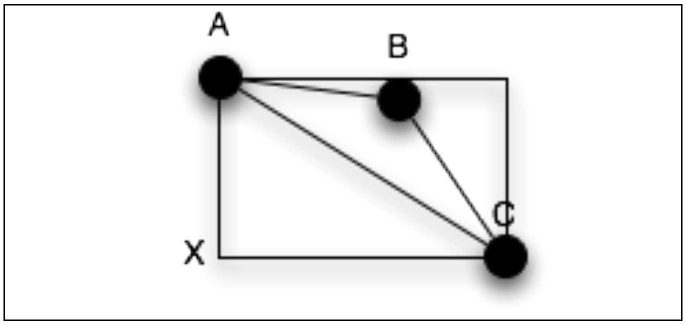


Figure 2. Area-based algorithm. Areas covered by trapezoid ABCX and ACX are compared to decide if B does provide additional information. The less the difference the more A,B and C are on a same line

C. *Image algorithm*

This algorithm is quite naive. The idea is just to take a photo of a visualization of the sketch and scale it, for reducing the size of the image.

The critical aspect of this algorithm is the time required to create the image in some format (PNG in our case) out from the sketch's points characterization of the sketch.

D. *Compression algorithm*

The last algorithm uses compressions for reducing the amount data required to represent the sketch. We used the JavaScript Objection Notation (JSON) for representing the data as an array of pairs (array of 2-dimentional arrays) representing the position of the points:

[[20.7, 10.2], [8.3, 15.4], ...]

Having this representation as text data, it is compressed using the ZIP algorithm.

IV. RELATED WORK

Similar work has been done in the computer graphics area to reduce the size of 3D meshes without information loosing, analyzing the information from the "sketch" (mesh) in order to decide which vertexes are good candidates to be removed. Two important works on this area are presented by Jones, Durand and Desbrun [18]; and Lee, Sweldens, Schröder, Cowsar and Dobkin [19].

The former work is based on a robust estimation of vertex positions and local first-order predictors, based on triangles of the mesh, to generate predictions about vertexes' positions, avoiding outliers and reducing the number of used vertexes. Also, given that predictors are based on the orientation of the tangent planes, they use Mollification [20] to improve the final vertex set, avoiding the mesh noise through preservation of the corners and smoothing the normal only.

The latter work uses hierarchical surface representations of 3D objects, removing a maximal independent set of vertices with low degree (number of edges adjacent to a certain vertex) using a priority queue based on both geometric and topological information. The algorithm randomly selects a non-marked vertex of degree less than 12, removes it and all adjacent edges from the vertex set, marks its neighbors as unremovable and iterates this until no further vertices can be removed.

Both algorithms require large amounts of computation, and therefore it would require a powerful CPU and large memory, so it is not currently possible to implement them in mobile devices. Therefore, we prefer to use simple but powerful algorithms, based on points distance and area, requiring less computational power

.

Figure 3. GUI of "X". The main area is for drawing the sketch. The selection bar at the bottom is used to select the the characterization or compression algorithm and the button to perform it over the sketch and gejnerate the statistics

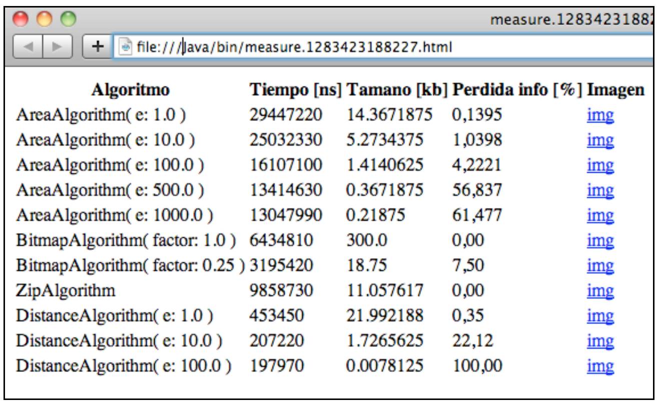
V. The "X" Application

The "X" application was developed using JavaSE in order to emulate the touch-screen of a mobile device with simplicity. The results of the statistics performed over the algorithms are stored on a webpage which is easy to export to a spreadsheet. The application is divided in four modules:

A. GUI module

With the graphical interface emulating a touch screen (Figure 3), the main class is in charge of creating the visual GUI, characterize the sketch using the available models and instantiate the algorithms implemented in the optimization module to be applied on the characterization.

Results produced after pressing the "generate" button are stored in a HTML page as shown in Figure 4.



Algoritmo	Tiempo [ns]	Tamano [kb]	Perdida info [%]	Imagen
AreaAlgorithm(e: 1.0)	29447220	14.3671875	0,1395	img
AreaAlgorithm(e: 10.0)	25032330	5.2734375	1,0398	img
AreaAlgorithm(e: 100.0)	16107100	1.4140625	4,2221	img
AreaAlgorithm(e: 500.0)	13414630	0.3671875	56,837	img
AreaAlgorithm(e: 1000.0)	13047990	0.21875	61,477	img
BitmapAlgorithm(factor: 1.0)	6434810	300.0	0,00	img
BitmapAlgorithm(factor: 0.25)	3195420	18.75	7,50	img
ZipAlgorithm	9858730	11.057617	0,00	img
DistanceAlgorithm(e: 1.0)	453450	21.992188	0,35	img
DistanceAlgorithm(e: 10.0)	207220	1.7265625	22,12	img
DistanceAlgorithm(e: 100.0)	197970	0.0078125	100,00	img

Figure 4. Webpage with results generated by the X application. The first collumn shows the algorithm applied and the parameter used (for example, the distance or area e), the second shows the time required to perform it, the fourth shows the size of the resulting characterization and the fourth one the information loss automatically calculated. The fifth collumn has a link to the reconstructed image

B. Models module

This is the application package where the models for the characterization of sketches are stored. Currently the only available characterization is the so called *polygonal* consisting of a set of points, but this might be changed if needed. Of course, if the model is changed the same optimization algorithms might not be applied.

C. Optimizations module

This package contains classes implementing the optimization methods. In the current version the four methods described are the only available; all of them can be applied to the polygonal representation model. In order to include a new method, the class that should implement it has to extend the *Algorithm* class, implementing *applyImpl* method that is called by the GUI with the Polygon object (the representation of the sketch) as parameter.

D. Statsmodule

This is the package where the statistical analysis classes and methods are stored, currently they include size, time and lost info percentage.

E. Application use

When the "generate" button of the interface is pressed after a sketch has already been drawn on the application's working space, the selected algorithm is performed and the statistics implemented on stats module are generated; they are stored in a HTML file as a table which include a graphical representation (PNG image) of the algorithm applied on the characterization (Figure 4). Moreover, the HTML is easy to export to spreadsheets to improve the data analysis.

VI. Analysis of Algorithms Performance

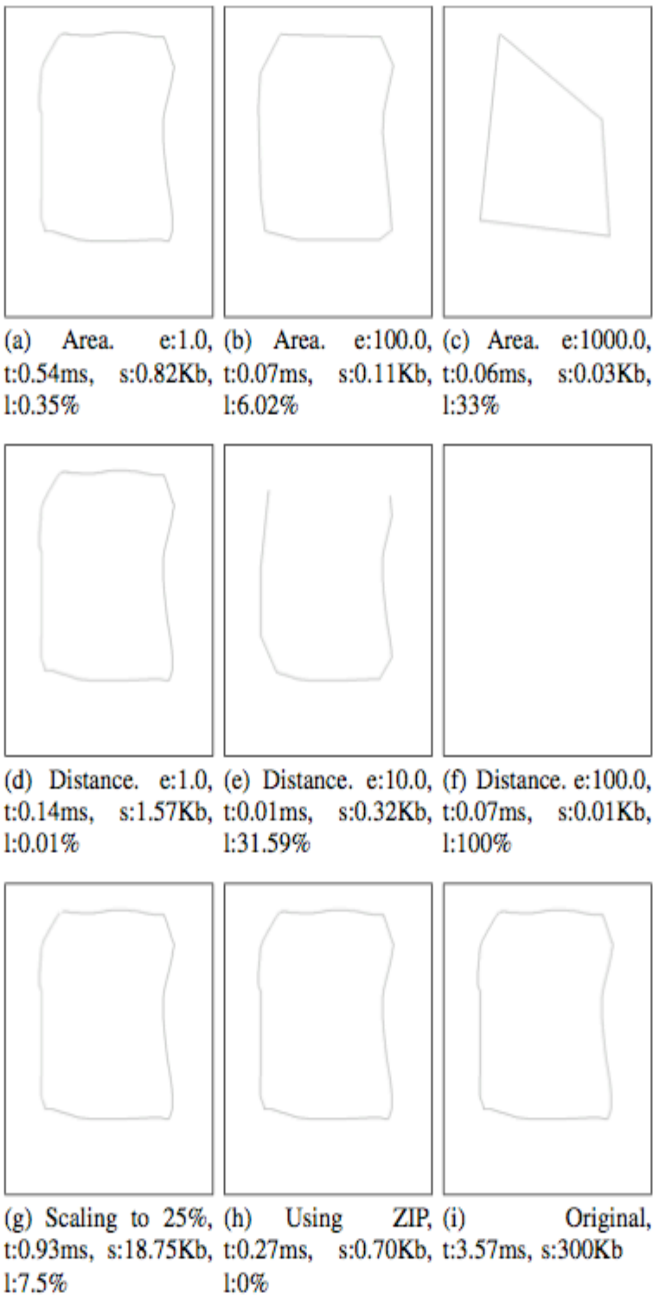
We focus our analysis on 3 properties:

- Size of the resulting characterization
- Time used by the algorithm on the characterization and/or compression of the sketch, and
- % of information lost due the characterization and/or compression.

The first dimension is related to the bandwidth needed to transmit the sketch from a mobile device, the second dimension is related to power needed by the computer device to perform the characterization and the third one is related to the quality of the characterization. We will study how they perform in basic, almost geometrical sketches and also in "advanced" sketches, containing more and less structured information. Our hypothesis is that on the first ones, the distance and area based algorithms will perform better, since a smaller number of points are needed to characterize them.

A. Basic sketches

As representatives for the basic figures we chose a freehand drawn box and a big *X* letter, in order to measure the performance (size, time and %) of the different algorithms over these two basic sketches. We compare the obtained values against the one we obtain by taking a "picture" of the sketch (creating a bitmap of the sketch) and send it through the network. Figure 5 shows the regenerated image of the box sketch after applying each algorithm one of the algorithms. The first row shows the image for the area algorithm using an ***e*** (difference in number of square pixels) of 1 (a), 100 (b) and 1000 (c) pixels. The value ***t*** corresponds to the time in milliseconds required by the algorithm to apply the algorithm to the original figure, ***s*** corresponds to the number of Kb of the characterization after applying the algorithm and ***l*** shows the percentage of similarity between the original and the transformed sketch calculated as described in section II. The second row shows the reconstructed images of the original sketch applying now the algorithm based on the area using an ***e*** (distance in number of linear pixels) of 1 (d), 100 (e) and 1000 (f). The third row shows the same when the original sketch is scaled down 25% (g), then when the ZIP algorithm is applied (h) and finally the original one (i), when a "photograph" of the original sketch is produced. Note that for basic sketches, even that the calculated percentage of lost information is around 30% in the worst case (without taking in consideration the case of total lost) the sketches are still recognizable for a human being.

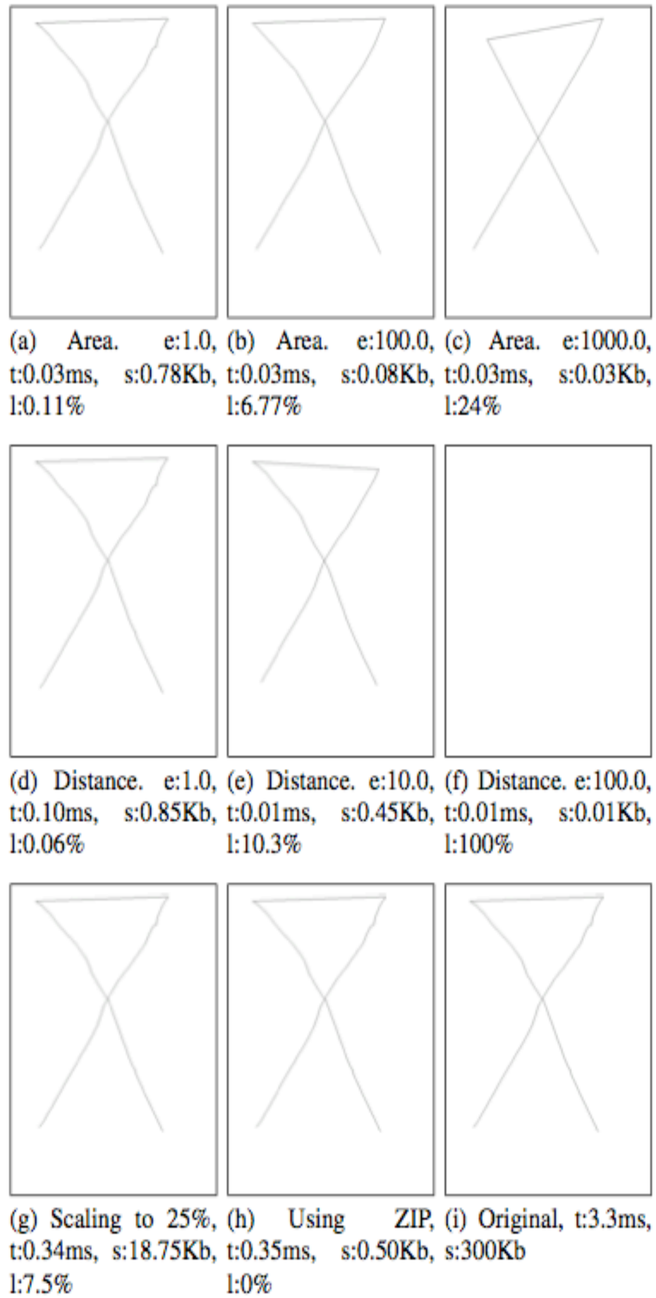


(a) Area. e:1.0, t:0.54ms, s:0.82Kb, l:0.35% (b) Area. e:100.0, t:0.07ms, s:0.11Kb, l:6.02% (c) Area. e:1000.0, t:0.06ms, s:0.03Kb, l:33%

(d) Distance. e:1.0, t:0.14ms, s:1.57Kb, l:0.01% (e) Distance. e:10.0, t:0.01ms, s:0.32Kb, l:31.59% (f) Distance. e:100.0, t:0.07ms, s:0.01Kb, l:100%

(g) Scaling to 25%, t:0.93ms, s:18.75Kb, l:7.5% (h) Using ZIP, t:0.27ms, s:0.70Kb, l:0% (i) Original, t:3.57ms, s:300Kb

Figure 5. Characterization of a box sketch



(a) Area. e:1.0, t:0.03ms, s:0.78Kb, l:0.11% (b) Area. e:100.0, t:0.03ms, s:0.08Kb, l:6.77% (c) Area. e:1000.0, t:0.03ms, s:0.03Kb, l:24%

(d) Distance. e:1.0, t:0.10ms, s:0.85Kb, l:0.06% (e) Distance. e:10.0, t:0.01ms, s:0.45Kb, l:10.3% (f) Distance. e:100.0, t:0.01ms, s:0.01Kb, l:100%

(g) Scaling to 25%, t:0.34ms, s:18.75Kb, l:7.5% (h) Using ZIP, t:0.35ms, s:0.50Kb, l:0% (i) Original, t:3.3ms, s:300Kb

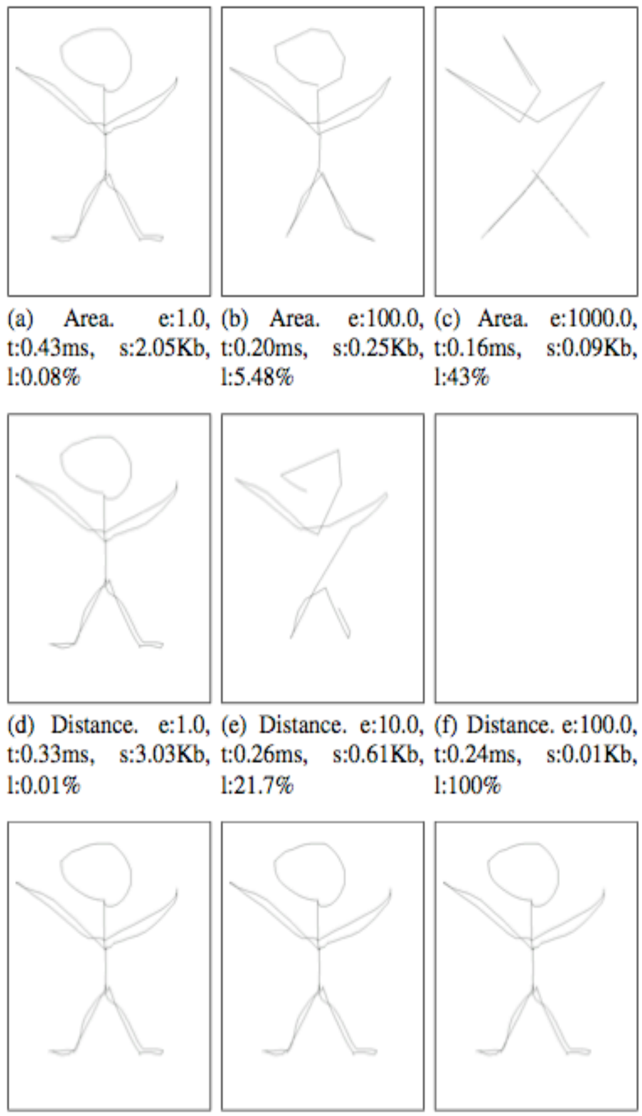
Figure 6. Characterization of a *X* sketch

B. *Advanced sketches*

The same experience was performed using more advanced sketches: a human figure (Figure 7) and a handwritten text message (Figure 8); calculating the same performance measures (size, time and %) for the various algorithms as done for the basic sketches.

As in the previous section, the order of the pictures is: using Area algorithm (a, b, and c), using Distance algorithm (d, e and f), scaling down the sketch (g), zipping the original sketch (h) and finally the original sketch (i).

From the results we can notice that for complicated sketches with information lost over 40% could produce a complete distortion of the original sketch; therefore algorithms with a high rate of information lost should be avoided.

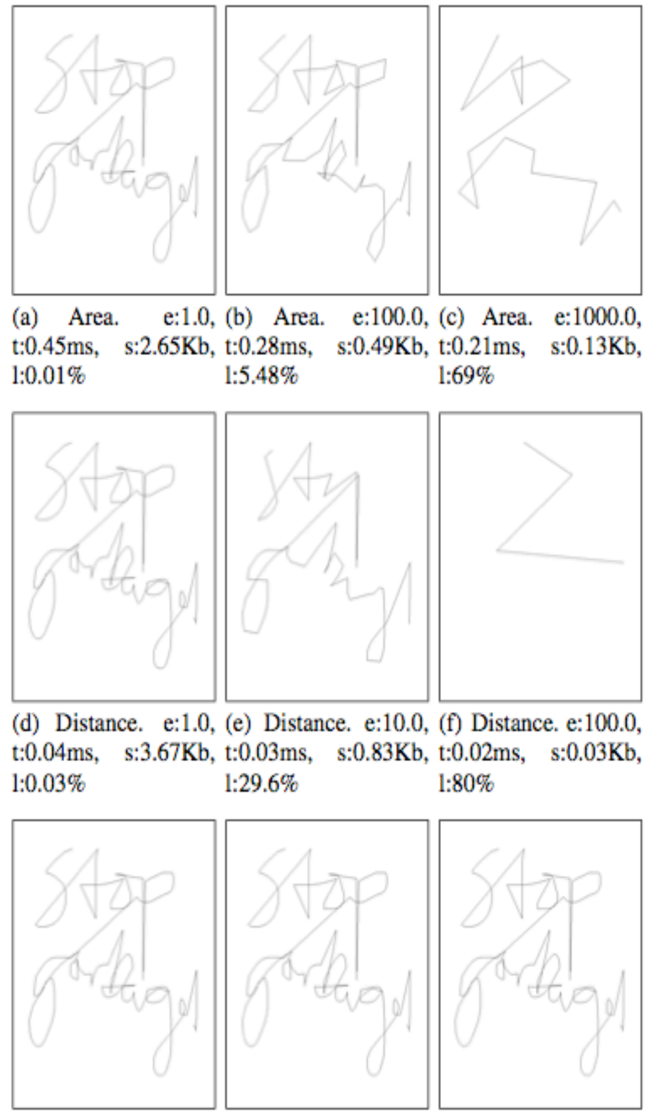


(a) Area. e:1.0, t:0.43ms, s:2.05Kb, l:0.08% (b) Area. e:100.0, t:0.20ms, s:0.25Kb, l:5.48% (c) Area. e:1000.0, t:0.16ms, s:0.09Kb, l:43%

(d) Distance. e:1.0, t:0.33ms, s:3.03Kb, l:0.01% (e) Distance. e:10.0, t:0.26ms, s:0.61Kb, l:21.7% (f) Distance. e:100.0, t:0.24ms, s:0.01Kb, l:100%

(g) Scaling to 25%, t:0.57ms, s:18.75Kb, l:7.5% (h) Using ZIP, t:0.83ms, s:1.74Kb, l:0% (i) Original, t:3.34ms, s:300Kb

Figure 7. Characterization of a *human figure* sketch



(a) Area. e:1.0, t:0.45ms, s:2.65Kb, l:0.01% (b) Area. e:100.0, t:0.28ms, s:0.49Kb, l:5.48% (c) Area. e:1000.0, t:0.21ms, s:0.13Kb, l:69%

(d) Distance. e:1.0, t:0.04ms, s:3.67Kb, l:0.03% (e) Distance. e:10.0, t:0.03ms, s:0.83Kb, l:29.6% (f) Distance. e:100.0, t:0.02ms, s:0.03Kb, l:80%

(g) Scaling to 25%, t:0.57ms, s:18.75Kb, l:7.5% (h) Using ZIP, t:0.98ms, s:1.74Kb, l:0% (i) Original, t:3.53ms, s:300Kb

Figure 8. Characterization of a *handwritten text message* sketch

C. *Results Analysis*

As seen in previous sections, algorithms with a high rate of lost information should be avoided. A summary with the data about the performances of the algorithms using basic sketches is presented in Table I, and the performance of the selected algorithms using advanced sketches is presented in Table II.

To validate the mathematical notion of "similar sketches" measured as the percentage of matching pixels between the original sketch and the resulting sketch after applying the algorithm and regenerating it we performed an opinion poll evaluating reconstructed sketches against their originals. The pool is available under `http://www.dcc.uchile.cl/~rcruzat/trabajoDirigido`. In this pool participants had to compare the original sketch with each one produced by the algorithms and give their subjective opinion about how similar they look by giving a "mark" between 1 (not similar at all) and 7 (look the same to me).

The pool was answered by around 100 people. All of them were computer science students with ages varying between 19 and 24 years old, 80% male. Results are presented in Figure 9. They were normalized to a number between 0 and 100 in order to more easily compare them with the percentage computed by the system.

Results show a strong correlation between our mathematical measure and the human perception for similarity between sketches. In fact, using area and distance algorithms with a parameter ***e = 1.0*** we obtain same results in terms of human perception than using scaling image and zip algorithm (which are lossless).

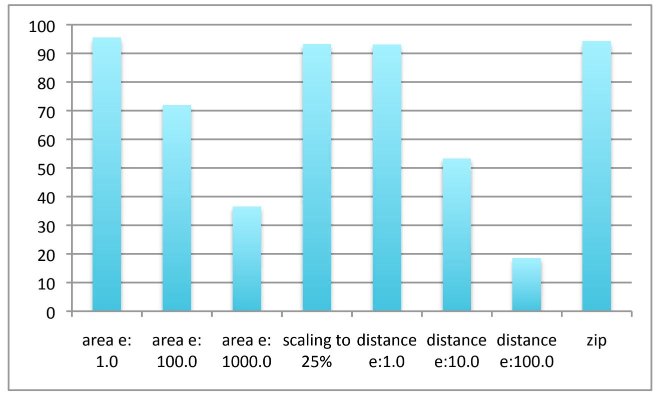


Figure 9. Similarity between sketches according to the subjective perception of the human eye. 100 means are quite similar.

TABLE I. PERFORMANCE OF ALGORITHMS WITH BASIC SKETCHES

Algorithms	***Time [msec]***	***Mean size [kb]***	***%***
Area ***e=1.0***	0.03-0.58	0.8	< 1%
Area ***e=100.0***	0.03-0.28	0.10	5-6%
Area ***e=1000.0***	0.03-0.06	0.03	> 24%
Distance ***e=1.0***	0.10-0.33	1.0	0.02%
Distance ***e=10.0***	0.10-0.33	0.39	0.02%
Scaling to 25%	0.34-0.93	18.75	7.5%
ZIP	0.28-0.98	0.6	---
Original	3.4	300	---

TABLE II. PERFORMANCE OF ALGORITHMS WITH ADV. SKETCHES

Algorithms	***Time [msec]***	***Mean size [kb]***	***%***
Area ***e=1.0***	0.03-0.58	2.2	< 1%
Area ***e=100.0***	0.03-0.28	0.35	5-6%
Area ***e=1000.0***	0.03-0.06	0.11	> 40%
Distance ***e=1.0***	0.10-0.33	3.5	0.02%
Distance ***e=10.0***	0.10-0.33	0.72	0.02%
Scaling to 25%	0.34-0.93	18.75	7.5%
ZIP	0.28-0.98	1.74	---
Original	3.4	300	---

VII. CONCLUSIONS AND FUTURE WORK

On this article we presented a study of sketch characterization and compression algorithms in order to apply them in mobile collaborative applications. Studied algorithms were based on distances, on areas, scaling the original figure, and ZIPing of the original figure, aiming to find the best characterization of sketches to be shared among users of a wireless peer-to-peer ad-hoc network.

The results obtained are quite promising, since they show that algorithms based on areas and distances with a low error parameter (e) are the best selection for collaborative mobile applications, having a compression lower than a 1% the size of the original image with an almost lossless sketch characterization, in terms of a distance matrix and human perception.

In this article we also introduce our Java Tool "X", a modular application used to test different algorithms for characterization of sketches; the main idea under the design of "X" was to build a modular application for testing that being easy to incorporate new algorithms and optimizations.

As a future work, we aim to continue studying:

- Algorithms for characterization and compression of sketches
- Applications for the use of sketches, as educational, working of idea-sharing applications presented in Section I; and,
- How to improve application "X" to auto-generate code for JavaME, Android and/or iphone, aiming to provide an easy way to test characterization/compression algorithms on a real mobile and wireless environment.

Application "X" can be downloaded and tested from `http://simula.inf.udp.cl/?p=179`.

ACKNOWLEDGEMENTS

Authors would like to thank to SIMULA Research Group, of the School of Engineering and Telecommunications of Universidad Diego Portales, Chile, for providing support and testing devices of the application “X”. Application “X” and its development on Android and i-phone are developed under license GPL by SIMULA Research Group.

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] G. Reitmayrand and D. Schmalstieg, “Mobile collaborative augmented reality,” in ISAR. Published by the IEEE Computer Society, 2001, p. 114.

[3] E. Ngai and A. Gunasekaran, “A review for mobile commerce research and applications”, Decision Support Systems, vol. 43, no. 1, pp. 3–15, 2007.

[4] C. Looi, D. Jonassen, and M. Ikeda, “Towards sustainable and scalable educational innovations informed by the learning sciences: sharing good practices of research, experimentation and innovation”. IOS Press Inc, 2005.

[5] Z. Pan, R. Aylett, H. Diener, X. Jin, S. Göbel, and L. Li, “Technologies for e-learning and digital entertainment” Springer-Verlag Berlin Heidelberg, 2006. ISBN 3-540334-238

[6] N. Baloian, F. Claude, R. Konow, and S. Kreft, “E-Breaker: Flexible,

distributed environment for collaborative authoring", in Proceedings of the 13th International Conference on Computer Supported Cooperative Work in Design. IEEE Computer Society, 2009, pp. 173-178.

[7] L. Kleinrock, “Nomadic computing and smart spaces,” Internet Computing, IEEE, vol. 4, no. 1, pp. 52–53, 2002.

[8] G. Zurita, N. Baloian, and F. Baytelman, “A col- laborative face-to-face design support system based on sketching and gesturing,” Advanced Engineering Informatics, vol. 22, no. 3, pp. 340–349, 2008.

[9] E. Lank and S. Phan, “Focus+ Context sketching on a pocket PC,” in CHI’04 extended abstracts on Human factors in computing systems. ACM, 2004, pp. 1275– 1278.

[10] A. May, V. Mitchell, S. Bowden, and T. Thorpe, “Opportunities and challenges for location aware computing in the construction industry,” in Proceedings of the 7th international conference on Human computer interaction with mobile devices & services. ACM, 2005, pp. 255–258.

[11] A. MacEachren, C. Guoray, I. Brewer, and J. Chen, “Supporting Map-Based geocollaboration through Nat- ural Interfaces to Large-Screen Displays,” Cartographic Perspectives, vol. 54, no. 1, pp. 4–22, 2006.

[12] P. Antunes, C. Sapateiro, G. Zurita, and N. Baloian, “Integrating Spatial Data and Decision Models in an E-Planning Tool,” in Collaboration and Technology: 16th International Conference, CRIWG 2010, Maastricht, The Netherlands, September 20-23, 2010, Proceedings. Springer, 2010, p. 97.

[13] N. Baloian, J. Pino, G. Pena, and G. Zurita, “Learning with patterns: An effective way to implement computer supported pervasive learning,” in Computer Supported Cooperative Work in Design (CSCWD), 2010 14th International Conference on. IEEE, 2010, pp. 677– 682.

[14] G. Zurita, N. Baloian, F. Baytelman, and M. Morales, “A gestures and freehand writing interaction based Electronic Meeting Support System with handhelds". On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE, pp. 679--696, 2006.

[15] J. Landay and B. Myers, “Sketching interfaces: Toward more human interface design”, Computer, vol. 34, no. 3, pp. 56–64, 2002.

[16] R. van der Lugt, “Functions of sketching in design idea generation meetings”, in Proceedings of the 4th conference on Creativity & cognition. ACM, 2002, pp. 72–79.

[17] Y. Li, J. Landay, Z. Guan, X. Ren, and G. Dai, “Sketching informal presentations”, in Proceedings of the 5th international conference on Multimodal interfaces. ACM, 2003, p. 241.

[18] T. R. Jones, F. Durand, and M. Desbrun, “Non-iterative, feature-preserving mesh smoothing,” ACM Trans. Graph., vol. 22, pp. 943–949, July 2003. [Online]. Available: http://doi.acm.org/10.1145/ 882262.882367

[19] A. Lee, W. Sweldens, P. Schro ̈der, L. Cowsar, and D. Dobkin, “MAPS: Multiresolution adaptive parame- terization of surfaces,” in Proceedings of the 25th an- nual conference on Computer graphics and interactive techniques. ACM, 1998, pp. 95–104.

[20] D. Murio, “The mollification method and the numerical solution of ill-posed problems. Wiley-Interscience, 1993.