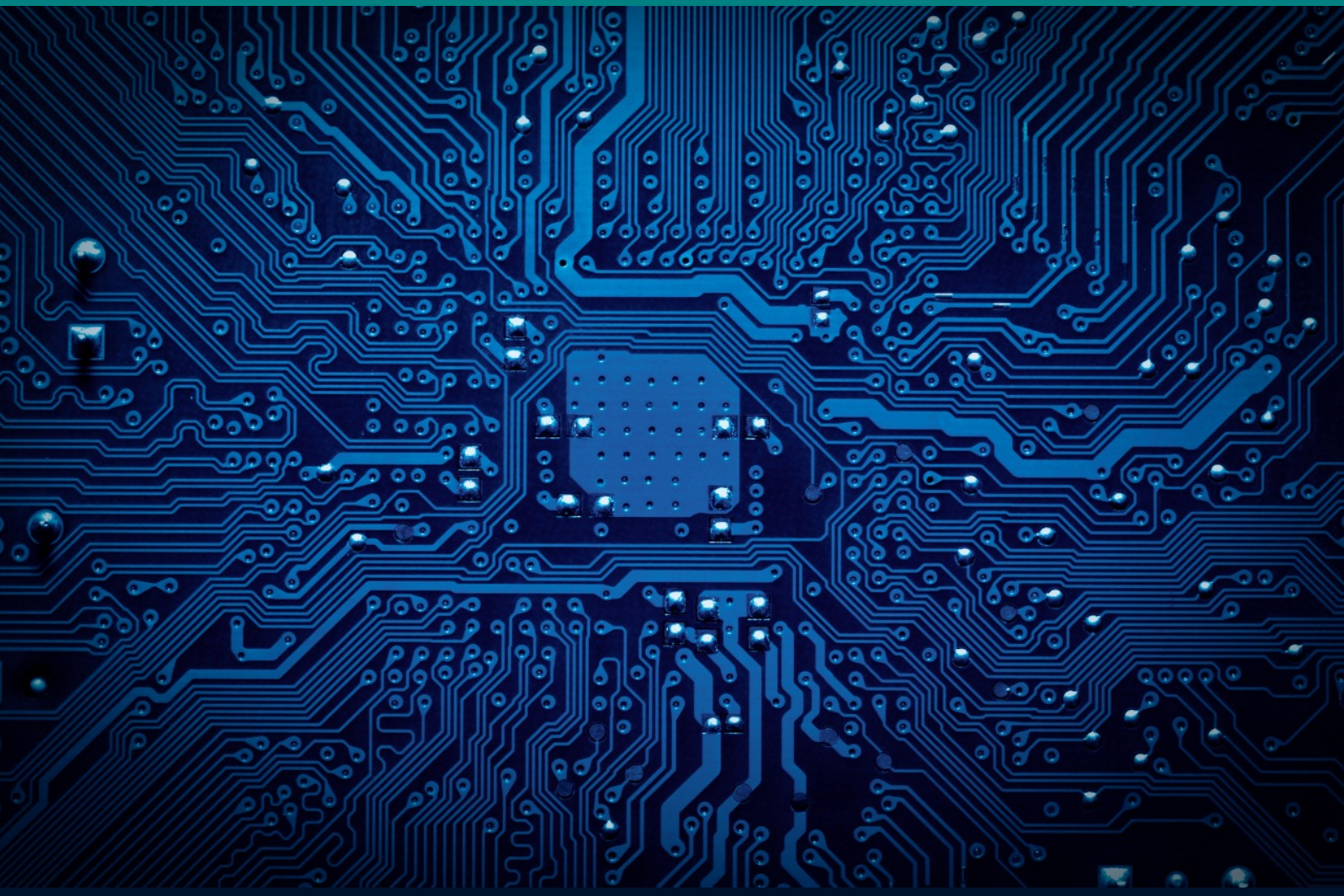
AJC
Academic Journal on Computing, Engineering and Applied Mathematics
EAM

2022
Volume 3 Issue 1

Academic Journal on Computing, Engineering and Applied Mathematics



UNIVERSIDADE FEDERAL DO TOCANTINS

ISSN: 2675-3588

Universidade Federal do Tocantins

Dados Internacionais de Catalogação na Publicação (CIP)

Academic Journal on Computing, Engineering and Applied Mathematics (AJCEAM) [recurso eletrônico] / Universidade Federal do Tocantins, Curso de Ciência da Computação. – vol. 03, n. 01 ([outubro/março], 2022) – Palmas - TO, UFT, 2022. ISSN nº 2675-3588.

Quadrimestral no primeiro ano de publicação 2020
Semestral.
Disponível em:
https://sistemas.uft.edu.br/periodicos/index.php/AJCEAM/index

1. Ciência da Computação - periódico. 2. Matemática Aplicada. 3. Computação Aplicada. 4. Engenharias. 5. Ciências Exatas. I. Universidade Federal do Tocantins.

CDD 22.ed. 004

Ficha Catalográfica elaborada por Edson de Sousa Oliveira – CRB/2 – 1069.

Expediente

Realização
Fundação Universidade Federal do Tocantins (UFT)
Quadra 109 Norte, Avenida NS-15, ALCNO-14 | Bloco III | sala 214 |Plano Diretor Norte | 77001-090 | Palmas / TO | Brasil

Periodicidade
Este periódico possui periodicidade semestral e utiliza a Licença Creative Commons 4.0 - CC BY-NC 4.0. Contudo, a publicação dos artigos em modalidade avançada ou ahead of print, ou seja, tão logo os manuscritos aprovados sejam editados para publicação, é possível. O AJCEAM não possui taxas de publicação, tanto pouco de submissão de manuscritos, sendo totalmente gratuita para autores e leitores.

Indexadores
Google Acadêmico, desde 9 de maio de 2020
International Standard Serial Number – ISSN, desde 28 de maio de 2020
Crossref, desde 7 de junho de 2020
Revistas de Livre Acesso – LivRe, desde 24 de junho de 2020

Sumário

Editorial (Português): Academic Journal on Computing, Engineering and Applied Mathematics

Warley Gramacho da Silva[1]

[1] *Universidade Federal do Tocantins, Palmas / TO, Brasil*

A equipe AJCEAM, em busca de mais visibilidade para o periódico e os autores, tem a satisfação de anunciar que o periódico está integrado com o ORCID[1] (*Open Researcher and Contributor ID*) através da sua atualização automática, ou seja, as informações do manuscrito para a qual o autor inclua o número ORCID, durante o processo de submissão, terá seu registro automático após a publicação. Sendo assim, a partir desta primeira edição de 2022, o AJCEAM passará a incluir o número ORCID dos autores em seus artigos.

Para esta edição o AJCEAM traz dois trabalhos com temáticas relevantes para computação e sua aplicação em problemas reais.

O primeiro trabalho, intitulado "*Exploring Super-Resolution for Face Recognition*", escrito por Araújo e Ribeiro [1], busca explorar diferentes cenários e situações envolvendo imagens, aprofundar os estudos na técnica de super resolução para o reconhecimento de face em imagens de baixa resolução, com o intuito de melhorar sua efetividade. O reconhecimento biométrico faz parte de muitos aspectos da sociedade moderna e trabalhos como esse são norteadores de pesquisas nesse contexto.

O segundo trabalho, Ribeiro, Carvalho e Almeida [2], através do artigo intitulado "*Optimization of state assignment in a finite state machine: evaluation of a simulated annealing approach*", propõe uma abordagem, via algoritmo *Simulated Annealing*, para resolução do problema de atribuição de estados em uma Máquina de Estados Finita. O trabalho explora um problema de otimização de hardware relacionado ao desenho industrial de circuitos. Como resultado, os autores trazem uma comparação de seus resultados com outros encontrados na literatura.

Por fim, agradecer aos autores em [1, 2] pela escolha do AJCEAM para terem seus trabalhos divulgados. Boa leitura e bons estudos.

Trabalhe arduamente e
nunca deixe de sonhar.
Goku (Dragon Ball)

[1] https://orcid.org/

REFERÊNCIAS

[1] P. Anderson Matias de Araújo and E. Ferreira Ribeiro, "Exploring super-resolution for face recognition," *Academic Journal on Computing, Engineering and Applied Mathematics*, vol. 3, no. 1, p. 1–8, out. 2021. [Online]. Available: https://sistemas.uft.edu.br/periodicos/index.php/AJCEAM/article/view/12616

[2] R. da Silva Ribeiro, R. Lima de Carvalho, and T. da Silva Almeida, "Optimization of state assignment in a finite state machine: evaluation of a simulated annealing approach," *Academic Journal on Computing, Engineering and Applied Mathematics*, vol. 3, no. 1, p. 9–16, dez. 2021. [Online]. Available: https://sistemas.uft.edu.br/periodicos/index.php/AJCEAM/article/view/12889

Exploring Super-Resolution for Face Recognition

Patrick Anderson Matias de Araújo[1] and Eduardo Ferreira Ribeiro[1]

[1] *Department of Computer Science, Federal University of Tocantins, 109 Norte, Av. NS 15, ALC NO 14, Palmas, Brazil*

Reception date of the manuscript: 07/07/2021
Acceptance date of the manuscript: 29/09/2021
Publication date: 19/10/2021

Abstract— Biometric recognition is part of many aspects of modern society. With the smartphones popularization, facial recognition gains space in this environment of biometric technologies. With the diversity of image capture devices, of different brands and qualities, the images will not always be in the ideal standard to be recognized. This article tests and compares different scenarios and situations to assess the results obtained by facial recognition in different environments. For this, the quantitative method of data analysis was used. In the first scenario, all images were submitted without changes. In the following, we have the reduction of image resolution, which may or may not be followed by enlargement to the original resolution via bicubic interpolation or through the Image Super-Resolution algorithm, these images can be all, or only that undergo tests. Results indicate that the first scenario obtained the best performance, followed by only the tests images change. The worst performance occurs where the properties of all images are affected. In situations where there is a reduction and enlargement optional, the enlargement option performs better, so the bicubic enlargement has an advantage over the ISR, the situation in which only the reduction occurs has the worst performance.

Keywords— Facial Recognition, Deep Learning, Super-Resolution, Images and Videos, Image Processing, Computer Vision

I. Introduction

Due to the Sars-CoV-2 pandemic, the adoption of digital resources has expanded and becoming more evident in many aspects of everyday life. In Brazil, for example, there were 15,632,584 cases and 436,001 deaths[1] due to COVID-19, after 447 days since the first national case, alternative solutions are needed, considering that, until May 18, 2021, vaccines were injected in 18.54%[2] of the population and are in short supply and sometimes missing due to the shortage of imported goods[3].

Even before this sad event, the paths were already paved: systems that monitor individuals through cameras, recognize their faces and activities, go beyond Chinese borders[4], a country well known for superb use of technologies, especially those that use biometric resources in surveillance of the population. Free of charge, mobile devices, social networks, and other services can recognize their users and even non-users.

Regardless of the current controversies in which the use of technologies has been present, its proper use can help in poignant issues, especially about issues such as the pandemic itself, an example are systems that can identify body temperature and prevent non-carriers of masks that do not enter public spaces[5], thus preventing the spread of the virus and, therefore, new infected ones. These technologies depend on concepts such as Deep Learning, biometrics, and super-resolution.

According to [1], with the popularization of smartphones, the adoption of biometric resources is observed in these devices to grant usage. [2] complement that facial recognition

Contact data: Patrick Anderson Matias de Araújo, patrick.araujo@uft.edu.br

[1]Data from May 18, 2021. (Monitor do coronavírus - saúde. **Folha de São Paulo**. Available at: <https://arte.folha.uol.com.br/equilibrioesaude/2020/casos-mortes-coronavirus-brasil-mundo/#/local/brasil>. Accessed on May 18, 2021.)

[2]Data from May 18, 2021. (Mapa da vacinação contra Covid-19 no Brasil | Vacina. **G1 - O portal de notícias da Globo (globo.com)**. Disponível em: <https://especiais.g1.globo.com/bemestar/vacina/2021/mapa-brasil-vacina-covid/>. Accessed on May 18, 2021.)

[3]CoronaVac vaccine is suspended due to lack of imported substances. (Declarações da gestão Bolsonaro contra a China afetam liberação de insumos de vacinas, diz Butantan. **Folha de São Paulo**. Available at: <https://www1.folha.uol.com.br/equilibrioesaude/2021/05/declaracoes-da-gestao-bolsonaro-contra-a-china-afetam-liberacao-de-insumos-diz-butantan.shtml>. Accessed on May 18, 2021.)

[4]A recent use of artificial intelligence for facial recognition occurred recently at Carnival in Salvador, which at the time helped to apprehend a criminal. (OH, S. Hyeon. Reconhecimento facial ajuda a prender criminoso no Carnaval de Salvador. **Canaltech**, São Bernardo do Campo, March 6, 2019. Available at: <https://canaltech.com.br/seguranca/reconhecimento-facial-ajuda-a-prender-criminoso-no-carnaval-de-salvador-134189/>. Accessed on: May 18, 2021.)

[5]A device has a system that can measure body temperature and identify a person who does not wear a mask. (ZARAMELA, Luciana. COVID-19: dispositivo indica quem está com febre em empresas ou instituições. **Canaltech**, São Bernardo do Campo, June 1, 2020. Available at: <https://canaltech.com.br/saude/covid-19-dispositivo-indica-quem-esta-com-febre-em-empresas-ou-instituicoes/>. Accessed on: May 18, 2021.)

is applied in different scenarios such surveillance, border patrol, and forensic science. Currently "[...] millions of images have been generated opening a range of possibilities for the most diverse purposes [...]" [3].

Facial recognition is understood as a set of techniques/algorithms that, through images, can recognize faces. Face data can be compared with data in a database to identify the individual with those characteristics. Facial recognition performed by machines is "[...] biometric techniques that consist in identifying facial patterns [...]" [4]. The main algorithms are named by their methodological approaches which can be holistic (based on appearance), structural (based on features) or hybrid (mixture of holistic and structural type). In the holistic type, there are the Eigenfaces, Fisherfaces and TensorFaces algorithms, in the structural type, the main algorithm is the Elastic Bunch Graph-Matching.

Deep Learning is a subfield of artificial intelligence. According to [5], this sub-area seeks, through its algorithms, to imitate the way the human brain works and its structure. A neural network in the context of Deep Learning is where the information goes through and gets processed, the first layer is the input layer, while the last layer it is called the output layer. A convolutional neural network (CNN) is a class inside Deep Learning and it is very used in many aspects of visual imagery.

Biometrics, on the other hand, is concerned with "[...] identifying people using their physical or behavioral characteristics" [5], to detect the bearer of the characteristic. This is done by transforming these characteristics into various statistical measures. An example might be fingerprint detectors in banks: a device that captures the biometric reading of an individual's hand and replaces the password in banking operations.

Super-resolution "[...] refers to the process of creating clear and high-resolution (HR) images from a single low-resolution (LR) image or from a sequence of low-resolution observations" [6]. An image that has undergone super-resolution processes allows for greater detail of the content displayed in the image. "The use of deep learning, specifically convolutional neural networks (CNNs) to perform the mapping between LR and HR images/patches have been extensively explored in recent years" [3].

In the context of digital images, "[...] interpolation is basically the process that uses known data to estimate values in points [...]" [7] not known.

Bicubic interpolation uses all 16 pixels (4x4 neighbors) closest to a pixel to make its estimate. Once these are at many ranges from the unspecified pixel, nearby pixels receive a bigger weighting in the computation. The resizing algorithms that use the bicubic interpolation logic carry out enlargement or reduction of the images. When compared to Bilinear and Nearest Neighbor interpolation, the Bicubic interpolation are more efficient and produces sharper images, that is why many image manipulation and edition programs implement the Bicubic interpolation.

It is noticeable that there is a symbiosis between the concepts. Normally, image resolution changes are performed using specific algorithms. However, an image obtained in low resolution can be transformed into a high-resolution image (super-resolution) using Deep Learning techniques and submitted to a facial recognition algorithm to detect biometric characteristics.

The super-resolution of images can be of paramount importance for the area of public security and commercial applications, as the images captured from conventional cameras and used in these areas are, according to [8], small and they differ from reduced low-resolution images from high-resolution images. This is occasioned by several factors, whether because the object of the captured images is far away, the environment does not help with lighting conditions, images that need to be reduced in their specificities to be transmitted in real-time, or the type of camera.

There is currently little research on super-resolution, therefore, this article proposes to deepen the studies of Deep Learning algorithms as well as super-resolution algorithms testing their integration to further increase their effectiveness in low-resolution images and videos. This project can bring new understandings about the subject discussed and, in the future, new applications in various areas, especially about public safety.

Specifically, this article will test different scenarios and situations involving images, their specifications and their content formed by pixels, to observe and choose a solution that best fits the criteria of effective and more accurate facial recognition, which makes the algorithm of face recognition correctly identify and recognize faces.

In an initial survey, authors [9] report that low-resolution images make face recognition a challenge. However, the researchers proposed a method to increase the resolution of images that differs from others that seek to investigate the direct relationship between high-resolution and low-resolution images. The researchers' method estimates high-frequency components that are not present in other methods.

The [10] research discusses the issue of invariant recognition of face posture using an MRF. An acronym that means Markov Random Field and is known for its computational complexity. For this, it was used the daisy descriptor for facial image representation in image recognition and created an implementation of a graphical processing unit of the MRF multiresolution matching process. Efficiency made the MRF approach viable and facilitates extensive empirical optimizations and evaluation studies.

It is proposed to transfer the pixel domain super-resolution reconstruction to face space with a smaller dimension in [11] research. This approach has the advantage of decreasing computational complexity regarding super-resolution image reconstruction, since the focus of the algorithm is not a visually improved image, but rather to search for the necessary information for the recognition system.

Paying attention to the impact of layer loss of artificial neural networks in the context of image processing, authors [12] show other alternatives for image restoration and, specifically, the importance of perceptually motivated losses when the resulting image is evaluated by a human observer. The performance of different types of losses is compared and a new differentiable error function is proposed. Furthermore, the authors show that image quality improves with more elaborate loss functions.

Finally, the research by [13] describes a system for recognizing and detecting human faces using real-time security cameras. The system uses the Viola-Jones face detector and then extracts local features and creates a shape-based feature

vector. The authors considered improving the performance and accuracy of the recognition and detection phases.

II. Materials and Method

As stated in the previous section, the aim of this study is to verify whether the use of super-resolution in low-quality images can improve the accuracy of a facial recognition algorithm. In this work, the quantitative method of data analysis was used with the support of the Face Recognition recognition algorithm, this algorithm uses the Python language and the dlib library. The choice of a solution that fits the criteria of effective facial recognition and better precision made this algorithm become the preference, in the algorithm documentation, it is said that the model has 99.38% accuracy[6].

For the tests to be carried out, a database widely used in the literature it is necessary for the optimal functioning of the algorithm.

A face database called Georgia Tech face database have been used for training and tests the image recognition system, which contains 750 images of 50 different people, each with 15 images of 640x480 resolution, totaling 128MB of images. These images were taken from photo sessions between 06/01/1999 and 11/15/1999 at the Center for Signal and Image Processing at the Georgia Institute of Technology. “The pictures show frontal and/or tilted faces with different facial expressions, lighting conditions, and scale”. [14].

These images are used by changing their properties, modifying their resolutions to compare different scenarios, and quantitatively evaluate these different situations. They are then only reduced or reduced and increased, the reduction occurs in a bicubic way, stretching happens or in a bicubic form, or using a super-resolution algorithm.

Classic algorithms work so that captured images go through a face detection process. Finally, face detection data are compared with data in a database to identify the subjects of the image in question as seen in Figure 1.

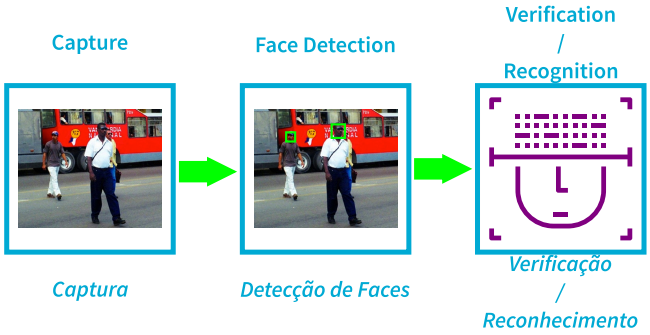


Fig. 1: Representation of how classical face recognition algorithms work.

The proposed system, represented in Figure 2, adds a step to the recognition process, which is the application of the super-resolution process to the images before their recognition in an attempt to improve image quality and, consequently, sophisticate the accuracy of the system of recognition.

[6]Documentation of the Face Recognition algorithm. (GEITGEY, Adam. **Face Recognition**: ageitgey/face_recognition. 2017. Available at: <https://github.com/ageitgey/face_recognition>. Accessed on Feb. 1, 2021.)

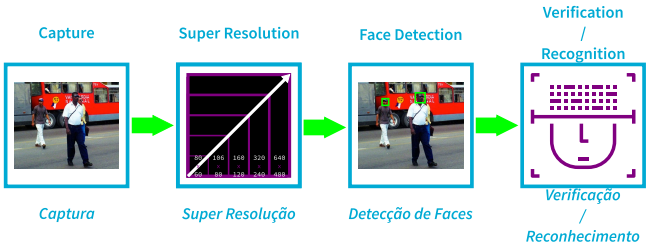


Fig. 2: Representation of the functioning of the proposed face recognition system.

The chosen algorithm needs, as input, images that will train the face recognition system for the tests to be executed in a controlled manner, in addition, to test images, that will be identified by the system. As part of the training of the images, they are found in an exclusive folder, where there are subfolders dedicated, each, for a single individual. As for the test images, they are also found in an exclusive folder, with no subfolders. In this context 200/750 images are dedicated to training and 550/750 images goes to the test process.

With this division done, the algorithm starts the training process and, when this part is finished, the detection of faces and recognition of identified faces begins.

Initially, the images were trained and tested according to the original specifications, keeping their resolutions.

Then, new directories were generated through the system, where each destination contains the same images from the database of faces, with the images contained in each directory of lower resolution. Each folder represents a resolution lower than the original resolution, as a result of dividing the original resolution by 2, 4, 6, and 8, resulting in resolutions 320x240, 160x120, 106x80, and 80x60, respectively. This means that both the images that are in the testing directory and the images that are in the training directory will have the same resolutions. However, it is possible that only the images found in the testing directory undergo this downsampling process, with the original resolution and properties being maintained in the images in the training directory.

Images that undergo the process of reducing their resolution can return to their original resolution, through bicubic interpolation or the Image Super-Resolution (ISR) algorithm. In this case, the ISR CNN is already trained and ready to use (also called in the literature off-the-shelf CNN). Both the techniques, algorithms and architecture are explained in the following.

The bicubic interpolation process is well known to be traditionally implemented in image manipulation applications, where the image is suppressed of detail making it blurry depending on how much the image is reduced.

Interpolation techniques constitute one of the components of Super-Resolution. Basically, with interpolation, new pixels are created from existing pixels. Although Super-Resolution uses interpolation, “say[...] interpolation techniques (nearest neighbor, bilinear and cubic convolution) differ from SR because in the first, only one image is used as a source of information for generate an image in higher resolution, different than what is used to produce an image using SR” [15].

Bicubic interpolation, uses all 16 pixels (4x4 neighboring pixels) closest to a pixel to perform its estimate. Since they are at various unspecified pixel ranges, nearby pixels

are given a higher weight in the calculation. Resizing algorithms that use bicubic interpolation logic perform enlargement or reduction of images. When compared to Bilinear and Nearest Neighbor interpolation, Bicubic interpolation is more efficient and produces sharper images, so many image manipulation and editing programs implement Bicubic interpolation.

On the other hand, images submitted through the ISR algorithm undergo more elaborate processing to recover details in low-resolution images. One of the objectives of this work is the generation of images from a super-resolution algorithm. From images that had their quality and specifications lowered, the super-resolution algorithm, namely Image Super-Resolution (ISR), makes the reconstruction of these images in high resolution. Then, these images will go through the face recognition algorithm to see if the use of super-resolution changes the effectiveness of facial recognition. The ISR project "[...] contains Keras implementations of different Residual Dense Networks for Single Image Super-Resolution (ISR), as well as scripts to train these networks using content and adversarial loss components" [16]. The choice of this solution is based on its availability, considering how expensive other solutions are, which implies adjusting the ideal settings for running the solution, installing libraries, etc. In the scope of super-resolution, we have what are called models; different implementations to obtain the super-resolution.

Specifically, for this work, the ISR network used is a off-the-shelf Convolutional Neural Network called Residual Dense Network for Image Super-Resolution [17]. Residual Dense Network extracts large local features through densely connected convolutional layers. RDN is composed of four elements, shallow feature extraction net (SFENet), residual dense blocks (RDBs), dense feature fusion (DFF), and the up-sampling net (UPNet).

This network architecture is formed by 6 convolutional layers stacked inside 20 Residual Dense Blocks (RDB) where each RDB contains 60 feature maps (of each convolutional layers inside the RDBs) and contains 64 feature maps for convolutions outside of RDBs (and of each RBD output). Also, this off-the-shelf network was pre-trained using the DIV2K Dataset [18] which contains 1000 super-resolution (2k) images with a great variety of contents. This network works very well with general images for Super-Resolution without any over-fitting during the training and testing phases.

These changes in image properties correspond to different scenarios to monitor the behavior of the face recognition algorithm. In real scenarios, the captured image may not be the best or there is an individual who is at a considerable distance even in images with considerable resolutions and properties. These techniques contribute to identification making the results more meaningful. In Figure 3 below, it is possible to observe in more detail the functioning of the proposed system.

The computer used to run the tests have 8 GB of RAM memory, 64 bits and an Intel®Core™ i7-7700 CPU @ 3.60GHz with 4 cores, 8 threads, and 8 MB de cache as its configuration. Specifically for the generation of super-resolution images due to the high computational cost, it became unfeasible to use the aforementioned computer, however, the Google Colab service was used, which allows outsourced processing, which made it necessary to send the images that would be performed in super-resolution to Google's proprietary cloud, namely Google Drive.

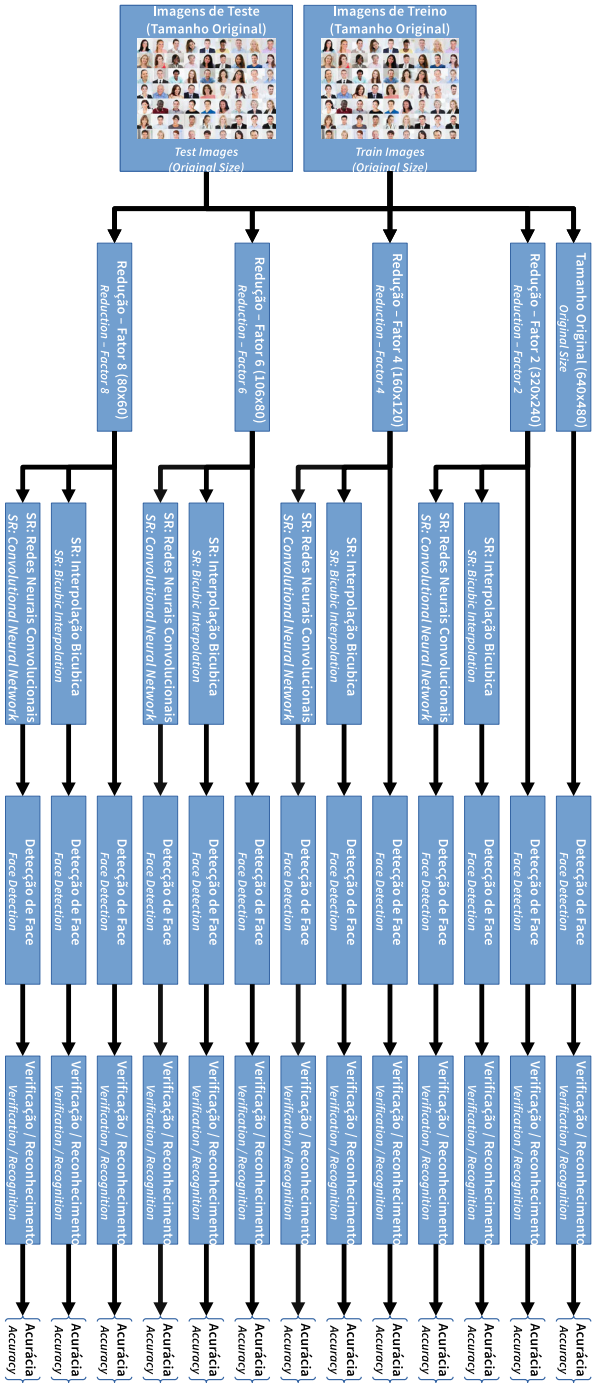


Fig. 3: Details of the operating of the proposed system.

III. Results

In this section, the results of all scenarios and situations mentioned in the previous part will be shown:

- Set of test images and training of original size;
- Reduction of original resolution (640x480) by factors of 2 (320x240), 4 (160x120), 6 (106x80) and 8 (80x60):
  - Sets of test and training images:

    - reduced by bicubic interpolation;
    - reduced and then increased to the original resolution by bicubic interpolation;
    - reduced by bicubic interpolation and then increased to the original resolution using the ISR algorithm.
- Only images from the test directory being retained the original properties of the training directory images:
    - Reduction of images of the testing directory by bicubic interpolation;
    - Reduction and then increasing the images of the test directory to the original resolution then by bicubic interpolation;
    - Reduction by bicubic interpolation and then enlargement of the test directory images to the original resolution using the ISR.

With the images without any change in their original specifications, in the first scenario, with regard to training and testing, 100% correctness was obtained in face identifications, not resulting in any error. That is, all 550 images were successfully identified. Figure 4 exemplifies the result obtained from a given input image.

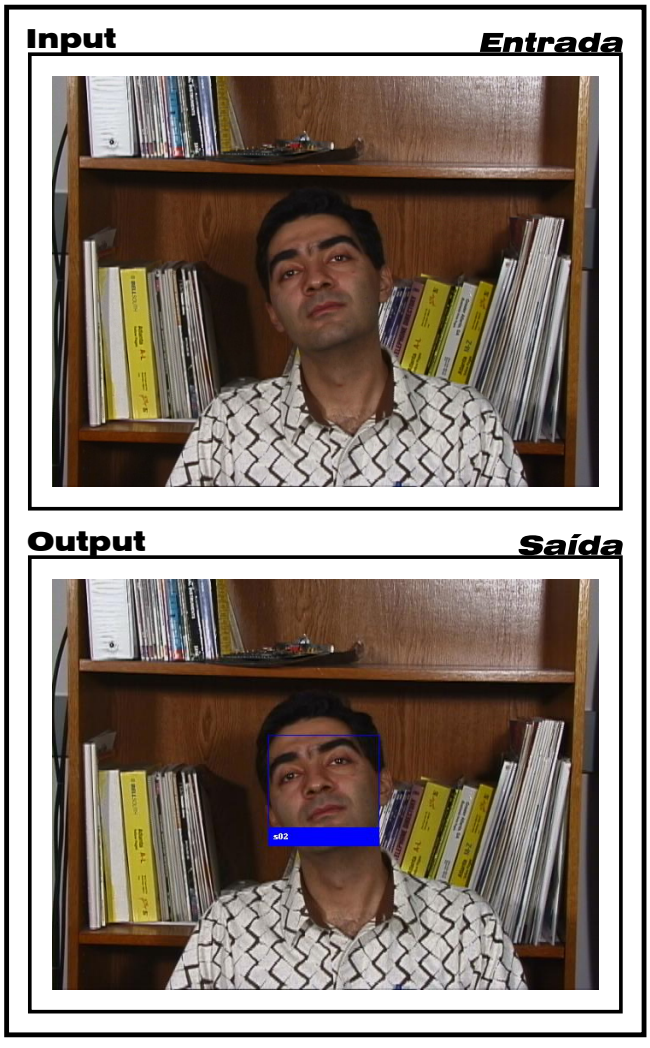


Fig. 4: Example of an image to be recognized and the recognition result.

Then, the scenario is changed, changing the image resolutions by dividing the original resolution by 2, 4, 6 and 8, as shown in Figure 5, the results vary.

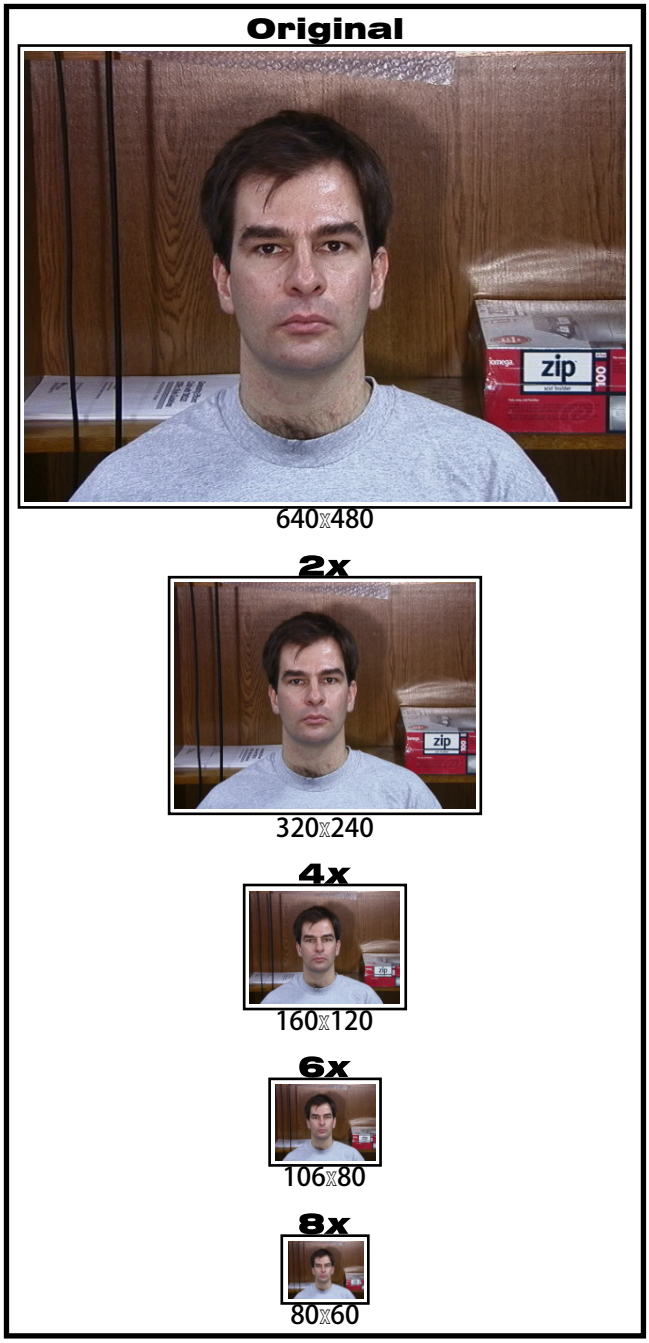


Fig. 5: Representation of an input image that goes through the bicubic interpolation process at a given lower resolution.

TABLE 1: RESULTS IN THE SCENARIO WHERE ALL IMAGES ARE BICUBICALLY DIMINISHED.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	548	2
4	160x120	543	7
6	106x80	17	553
8	80x60	0	550

Initially, at 320x240 resolution, resulting from the original resolution divided by 2, 548 images are correctly identified,

resulting in 99.64% of the total 550 images, leaving 0.36% of the images that result in error, that is, two images. By factor 4 (160x120), 543 images result in correct answers (98.73%), remaining seven images that result in error (1.27%). About factor 6 (106x80), 17 images are correctly identified, that is, only 3.09% of the images result in correct answers while the majority of 96.91% of the images result in an error (533 images). Finally, no image is corrected, with all images resulting in error.

Differently from the previous situation, where only the resolution of the images was reduced, there is also a situation in which the images are reduced and, through bicubic interpolation, they are increased to the original resolution, thus losing the original properties. Figure 6 illustrates this process.

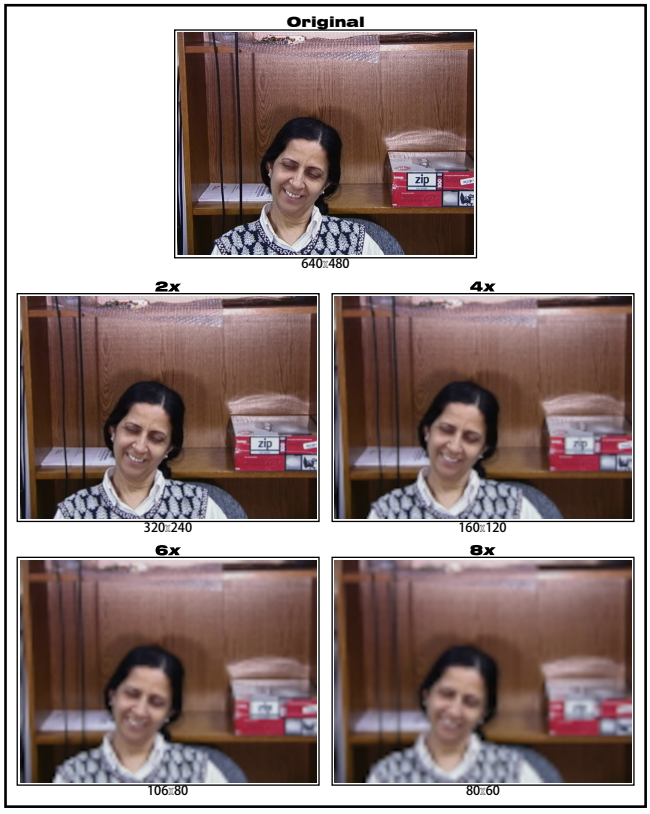


Fig. 6: Illustration of an image that is reduced and later enlarged by the interpolation process.

TABLE 2: RESULTS IN THE SCENARIO WHERE ALL IMAGES ARE DIMINISHED AND THEN BICUBICALLY ENLARGED.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	550	0
4	160x120	546	4
6	106x80	497	53
8	80x60	370	180

In this context, at 320x240 resolution (factor 2), it shows 100% accuracy of the total of 550 images. By factor 4 (160x120) it is seen that 546 (99.27%) of the images resulted in correct answers and only four images (0.73%) in error. Thus, by factor 6 (106x80), 90.36% (497 images) of the images result in correct answers while 9.64% (53 images) result in an error. Finally, there is the factor 8 (80x60) where 67.27% (370 images) of the images are correct and 32.73% (180 images) result in an error.

In comparison with the images that do not go through the bicubic stretching (as seen in Table 2), it can be seen as an improvement. As the factors increase, so performs the images that are enlarged over images that are only downsampled. By factor 2 there is an 0.36% improvement, in factor 4 the difference goes by 0.54%, in factor 6 there is the best performance by 87.27%, and finally, in factor 8 there are 67.27% of improvement.

Now it is introduced the ISR algorithm. In this situation, the images are bicubic reduced and then increased to the original resolution with the ISR algorithm. Figure 7 characterizes this process.

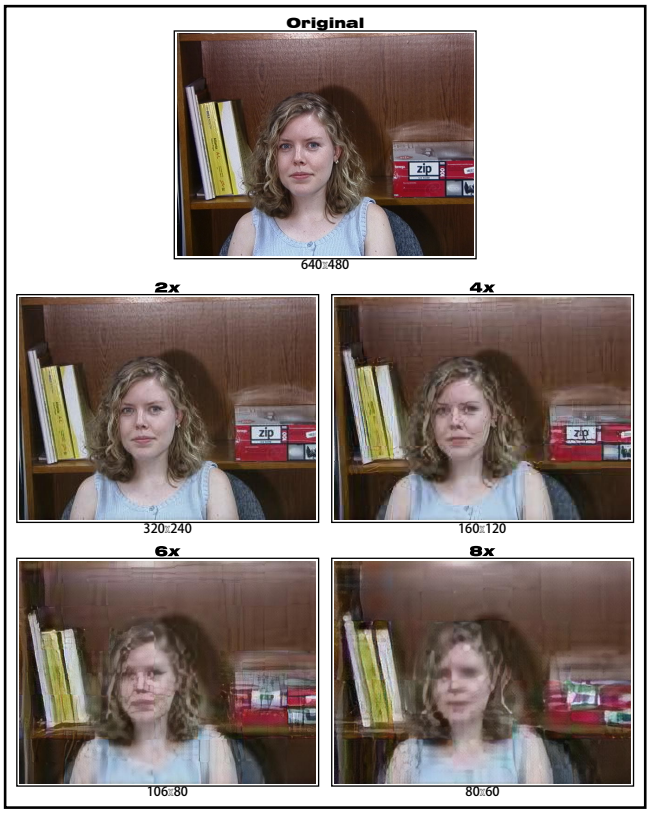


Fig. 7: Demonstration of an image that is reduced via bicubic interpolation and immediately increased to the original resolution using the ISR algorithm.

TABLE 3: RESULTS IN THE SCENARIO WHERE ALL IMAGES ARE BICUBICALLY DIMINISHED AND THEN AUGMENTED USING THE ISR ALGORITHM.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	550	0
4	160x120	531	19
6	106x80	370	130
8	80x60	181	369

As it can be seen in Table 3, the results in this scenario with the resolution of 320x240 (factor 2), it shows 100% accuracy of the total of 550 images. By factor 4 (160x120), it can be seen that 531 (96.5%) of the images resulted in correct answers and 19 images (3.5%) in error. In factor 6

(106x80) 67.27% (370 images) of the images result in correctness while 23.63% (130 images) result in an error. Finally, there is the factor 8 (80x60) where 32.9% (181 images) of the images are correct and 67.1% (369 images) result in an error.

Ultimately, in the last scenario, only the images located in the test directory are resized, keeping the properties of the images in the training directory, this comes close to reality and in most applications. In real environments the images are controlled and captured initially when a person is registered in the database, so these images need to be in good resolutions.

TABLE 4: RESULTS IN SCENARIO WHERE ONLY TEST DIRECTORY IMAGES ARE SHRUNK.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	550	0
4	160x120	549	1
6	106x80	110	440
8	80x60	0	550

As seen in the Table 4, starting with factor 2 of 320x240 resolution, the results achieved through the original resolution are maintained, there are 100% correct answers in the total of 550 images. By factor 4 of 160x120 resolution, the results change subtly, there is an error which corresponds to 0.2% of the 550 images, thus, 99.98% of the images result in correctness, which corresponds to 549 images. On the other hand, in factor 6 of 106x80 resolution, there is a considerable deterioration concerning correct answers, resulting in 80% of errors (440 images) and 20% (110 images) of correct answers. Finally, with factor 8 of 80x60 there is the worst performance, it does not result in any accuracy and with 100% (550 images) of images identified wrongly. In comparison with the Table 1 the results in this scenario proves an enhancement. This also means that high quality and resolution train images tends to improve the outcome, but for extreme low quality test images neither situation are shown as ideal. By factor 2 we have an 0.36% improvement, in factor 4 the difference goes by 1.09%, in factor 6 there are the best performance by 16.9%, and finally in factor 8 there are no improvement, in both cases no image was recognized.

Similarly, the previous situation, now in addition to decreasing the resolution, through bicubic interpolation, the images are increased to the original resolution, thus losing the original properties.

TABLE 5: RESULTS IN THE SCENARIO WHERE ONLY TEST IMAGES ARE REDUCED AND THEN ENLARGED.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	550	0
4	160x120	545	5
6	106x80	527	23
8	80x60	399	151

As it is shown in Table 5, initially, at factor 2 (320x240), there is 100% correctness (550 images) and no errors. Continuing by factor 4 (160x120), there is 99.1% correctness (545 images) and five images resulting from error. By factor 6, the images errors increase, resulting in 4.2% of the images (23 images) with 95.8% of the images correctly identified. Finally, with factor 8, we have that 72.5% of the images result in correct answers (399) and 27.5% of the images (151 images) correspond to errors. The comparison between Table 2 and 5 were both satisfy similar conditions, exhibit a better performance Table 2. By factor 2 both recognizes all images. In fator 4, the Table 2 takes advantage by 2.72%. Factor 6 there are 23.09% of difference. In the last factor, factor 8 there are 32.9% of variation. This means that the use of high quality images in the training does not improve the performance over images that are diminished and in sequence enlarged.

In the last situation of this scenario, the images of the tests and training directories were reduced and increased through the Image Super-Resolution algorithm, as seen in Table 6.

TABLE 6: RESULTS IN THE SCENARIO THAT TEST IMAGES ARE BICUBICALLY DIMINISHED AND THEN AUGMENTED USING THE IMAGE SUPER-RESOLUTION ALGORITHM.

Configuration		Results	
Factor	**Resolution**	**Hits**	**Errors**
2	320x240	550	0
4	160x120	544	6
6	106x80	432	118
8	80x60	234	316

The calculation in this configuration returns in factor 2 100% correctness. In factor 4 we have 544 correct answers (98.9%) and 6 errors (1.1%). With factor 6, the computation points to 432 correct answers (78.55%) and 118 errors (21.45%). Lately, in factor 8, there are 234 correct answers (42.55%) and 316 errors (57.45%). Comparing Table 3 and the Table 6 significant changes are shown in favor of Table 6. By factor 2 both recognizes all images. In factor 4 Table 6 shows an improvement of 2.36%. Factor 6 this difference goes by 11.27%. At last, in factor 8 are shown an improvement of 9.63%. This means that in the training process when the images are bicubically diminished and then enlarged using the ISR algorithm the results are better than when all images go through this process. But they are overcome by the results obtained by enlargement by bicubic interpolation.

The Figure 8 presents the summary of the main results by the number of correct answers of face recognition comparing the four main configurations: resampling the test and training images using bicubic interpolation (a) and super-resolution via CNN (b), and resampling only test images using bicubic interpolation (c) and super-resolution via CNN (d) for the four factors (2, 4, 6 and 8). Note that in all settings, reducing the image by half the size, the number of hits remains the same, ie 100 percent. As the resolution starts to decrease, the number of hits also decreases. In all cases the super-resolution using bicubic interpolation performed better than CNN mainly for the lowest factor when the image is almost all out of focus.

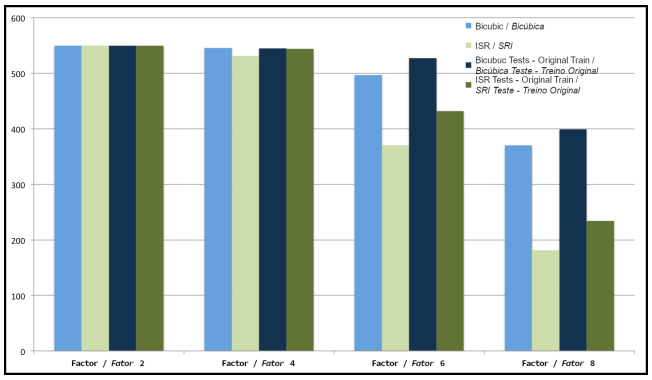


Fig. 8: Summary of Results in number of hits of the recognition process grouped in different test configurations by the two super-resolution algorithms.

IV. Conclusions

Given the results in different scenarios and situations, it is possible to observe that the ideal situation is that of images at high resolutions where the best results are obtained. However, in everyday situations, it is not always possible to rely on scenarios where there are images of excellent quality and resolutions.

With this in mind, even with images at low resolutions, the process of increasing the resolution considerably improves the results and performance of the algorithm in question.

The data indicate that, with the images that are trained in high resolution, even with the images that will be identified in lower resolution, the results are superior to the scenarios in which the training takes place with images in lower resolutions, which is the case of many applications used daily.

There was an expectation that the super-resolution would improve and enhance the results in terms of accuracy, which did not bring concrete results in this regard. Although visually, according to images that undergo super-resolution, as long as they have not very low resolutions, they show greater detail, which does not necessarily make the performance of face recognition more significant.

The bicubic interpolation used to increase images to the original resolutions with loss results in superior outcomes and, therefore, its use is justified in this context. Even without any new information, the interpolation uses the image's original pixels. The best result obtained in the interpolation process can be related to the chosen face recognition algorithm, which is based on fine detail. Deep Learning algorithms generate several artifacts that can confuse the recognition algorithm.

In future works, other facial recognition algorithms can be tested so other neural networks. Bicubic interpolation in comparison with Deep Learning can be cheaper and a faster alternative for low-budget systems that generate small quality and resolution images, improving the accuracy of results in a considerable way.

V. Acknowledgment

First I would like to thank the Federal University of Tocantins (*UFT*), the Scientific Initiation Voluntary Institutional Program (*PIVIC*), and Eduardo Ferreira Ribeiro the greatest advisor anyone can have.

I would like to thanks for the paramount help and assistance Eder Ahmad Charaf Eddine and Luciano de Jesus Gonçalves.

References

[1] H. Hofbauer, L. Debiasi, and A. Uhl, "Mobile face recognition systems: Exploring presentation attack vulnerability and usability," in *2019 International Conference on Biometrics (ICB)*, 2019, pp. 1–7.

[2] S. Kirchgasser, A. Uhl, Y. Martinez-Diaz, and H. Mendez-Vazquez, "Is warping-based cancellable biometrics (still) sensible for face recognition?" in *2020 IEEE International Joint Conference on Biometrics (IJCB)*, 2020, pp. 1–9.

[3] E. Ribeiro, A. Uhl, and F. Alonso-Fernandez, "Iris super-resolution using cnns: is photo-realism important to iris recognition?" *IET Biom.*, vol. 8, no. 1, pp. 69–78, 2019. [Online]. Available: https://doi.org/10.1049/iet-bmt.2018.5146

[4] T. D. Bissi, "Reconhecimento facial com os algoritmos eigenfaces e fisherfaces," Uberlândia, p. 41, 2018.

[5] K. Sundararajan and D. L. Woodard, "Deep learning for biometrics: A survey," *ACM Comput. Surv.*, vol. 51, no. 3, pp. 65:1–65:34, 2018. [Online]. Available: https://doi.org/10.1145/3190618

[6] R. R. Schultz and R. L. Stevenson, "Extraction of high-resolution frames from video sequences," *IEEE Trans. Image Processing*, vol. 5, no. 6, pp. 996–1011, 1996. [Online]. Available: https://doi.org/10.1109/83.503915

[7] W. B. Dourado, "Avaliação de técnicas de interpolação de imagens digitais," Master dissertation, Universidade Estadual Paulista (UNESP), Universidade Estadual Paulista (UNESP), Faculdade de Ciências e Tecnologia, Presidente Prudente, Aug. 2014. [Online]. Available: http://hdl.handle.net/11449/115892

[8] X. Xu, W. Liu, and L. Li, "Low resolution face recognition in surveillance systems," *Journal of Computer and Communications*, vol. 02, pp. 70–77, 01 2014. [Online]. Available: https://doi.org/10.4236/jcc.2014.22013

[9] E. Bilgazyev, B. A. Efraty, S. K. Shah, and I. A. Kakadiaris, "Improved face recognition using super-resolution," in *2011 IEEE International Joint Conference on Biometrics, IJCB 2011, Washington, DC, USA, October 11-13, 2011*. IEEE Computer Society, 2011, pp. 1–7. [Online]. Available: https://doi.org/10.1109/IJCB.2011.6117554

[10] S. R. Arashloo and J. Kittler, "Fast pose invariant face recognition using super coupled multiresolution markov random fields on a GPU," *Pattern Recognit. Lett.*, vol. 48, pp. 49–59, 2014. [Online]. Available: https://doi.org/10.1016/j.patrec.2014.05.017

[11] B. K. Gunturk, A. U. Batur, Y. Altunbasak, M. H. H. III, and R. M. Mersereau, "Eigenface-domain super-resolution for face recognition," *IEEE Trans. Image Processing*, vol. 12, no. 5, pp. 597–606, 2003. [Online]. Available: https://doi.org/10.1109/TIP.2003.811513

[12] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Computational Imaging*, vol. 3, no. 1, pp. 47–57, 2017. [Online]. Available: https://doi.org/10.1109/TCI.2016.2644865

[13] M. Davis, S. Popov, and C. Surlea, "Real-time face recognition from surveillance video," in *Intelligent Video Event Analysis and Understanding*, ser. Studies in Computational Intelligence, J. Zhang, L. Shao, L. Zhang, and G. A. Jones, Eds., 2011, vol. 332, pp. 155–194. [Online]. Available: https://doi.org/10.1007/978-3-642-17554-1_8

[14] "Georgia tech face database." [Online]. Available: http://www.anefian.com/face_reco.htm

[15] M. T. Jr and A. S. Rosa, "Super-resolução de imagens cbers 2," *XIII Simpósio Brasileiro de Sensoriamento Remoto*, pp. 1197–1204, 2007. [Online]. Available: http://marte.sid.inpe.br/col/dpi.inpe.br/sbsr@80/2006/11.16.00.47/doc/1197-1204.pdf

[16] F. C. et al., "Isr," https://github.com/idealo/image-super-resolution, 2018.

[17] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," 2018.

[18] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

Optimization of state assignment in a finite state machine: evaluation of a simulated annealing approach

Reinaldo da Silva Ribeiro[1], Rafael Lima de Carvalho[1] and Tiago da Silva Almeida[1,2]

[1] *Universidade Federal do Tocantins, Department of Computer Science, Palmas / TO, Brazil*
[2] *State University of Campinas, Institute of Computing, Campinas / SP, Brazil*

Reception date of the manuscript: 18/08/2021
Acceptance date of the manuscript: 16/11/2021
Publication date: 17/11/2021

Abstract— In this research, the application of the Simulated Annealing algorithm to solve the state assignment problem in finite state machines is investigated. The state assignment is a classic NP-Complete problem in digital systems design and impacts directly on both area and power costs as well as on the design time. The solutions found in the literature uses population-based methods that consume additional computer resources. The Simulated Annealing algorithm has been chosen because it does not use populations while seeking a solution. Therefore, the objective of this research is to evaluate the impact on the quality of the solution when using the Simulated Annealing approach. The proposed solution is evaluated using the LGSynth89 benchmark and compared with other approaches in the state-of-the-art. The experimental simulations point out an average loss in solution quality of 11%, while an average processing performance of 86%. The results indicate that it is possible to have few quality losses with a significant increase in processing performance.

Keywords— Finite State Machine. Simulated Annealing. Digital Systems. Metaheuristic.

I. Introduction

Hardware optimization demands a lot of research, not only from the professionals responsible for developing the hardware but also from physicists and chemists who can find ways or elements that improve the functioning of the hardware.

Regardless of the application, every hardware project is complex, demanding a great intellectual effort and, consequently, monetary. The process of design and development of a hardware component has several steps. In this work, the focus is on the optimization of Finite State Machines (FSM).

FSMs are abstractions of the behavior of a given circuit, whether it is a part of the whole of an Application-Specific Integrated Circuit (ASIC) or a conventional processor. When thinking in terms of algorithm, it is referred to the sequence of commands, or steps, that in a certain order performs a task.

This algorithm can be abstracted in the form of a machine where each step is represented by a state. Conventional computers allow us to perform only one step at a time and the transition between states is made through external or internal stimulus (inputs).

From this representation, it is possible to provide a physical model, where this FSM model can be synchronous about to with concerning the internal or external behavior, as well as it can be asynchronous, varying according to the application.

The optimization of an FSM can lead to a reduction in the physical size of the final circuit, resulting in savings in the critical path, area, and power. For the optimization of the FSM, the goal is composed of finding the best allocation of states and minimizing the size of the Boolean expressions that represent the machine behavior.

This is not a recent research topic [1, 2], however, due to its importance and being an NP-Complete[3] problem, is still an open topic because of breaking down of Dennard's law [4], which states that as the dimensions of a device go down, so does the power consumption. And many complex metaheuristic algorithms have been tested for this problem, such as Evolutionary Algorithms [5, 6], Tabu Seach [7], and Simulated Annealing [8, 9, 10].

As far as it is known, Ahmad et al [10] have proposed a complex hybrid method combining Genetic Algorithms with Simulated Annealing, to find optimal state-machine allocations. Thus, arises the question **how distant is the result with a much simpler and faster metaheuristic, which uses less computational resources (without a population of solutions)?** Therefore, the objective of this investigation is to provide an answer to this question. Furthermore, the main contribution of this paper is **the evaluation of the state assignment in a finite state machine solution produced by a simulated annealing algorithm**.

The remaining text is organized as the basic definitions about the considered problem are presented in Section IIa.

Contact data: Reinaldo da Silva Ribeiro, reinaldorib28@gmail.com

Furthermore, the Simulated Annealing is reviewed in Section IIb. The experimental setup, as well as the SA algorithm, is presented in Section III. In addition, the results are discussed in Section IV. Finally, the conclusions and future research directions are shown in Section V.

II. DEFINITIONS

a. Finite State Machine

Sequential circuits can be defined as circuits with a section made of combinational logic and another section of memory which are normally flip-flops. Where each stage that the sequential circuit advances are called a state. In each state, the circuit stores the inputs passed to define its output, and the state transition only occurs with the clock variation [11, 12].

An FSM has a finite number of inputs, constituting the set of $N = \{N_1, N_2, ..., N_n\}$. Thus, the circuit has a finite number of outputs, determined by the set of $M = \{M_1, M_2, ..., M_m\}$. The value contained in each memory element is called state variables, forming the set of $K = \{K_1, K_2, ..., K_k\}$. The values contained in the K memory elements define the current state of the machine. The internal transition functions generate the next state set $S = \{S_1, S_2, ..., S_s\}$, which depend on the inputs N and the current states K of the machine and are defined through combinational circuits . The values of S, which appear in the state machine transition function at time t, determine the values of the state variables at time $t+1$, and therefore define the next state of the machine.

The behavior of an FSM can be described through a state transition diagram or a state transition table. A state transition diagram or state transition table lists the current state, next state, input, and output. A state transition table has 2^N columns, one for each occurrence of the input set and 2^K rows, one for each occurrence of the state set.

The transition diagram is an oriented graph, where each node represents a state, and from each node emanate p oriented edges corresponding to the state transitions. Each oriented edge is labeled with the input that determines the transition and the output generated. FSM determine the next state $K(t+1)$, based only on the current state $K(t)$ and the current input $N(t)$. FSM can be represented by,

$$K(t+1) = f[K(t), N(t)] \tag{1}$$

where f is a state transition function. The output value $M(t)$ is obtained by,

$$M(t+1) = g[K(t)] \tag{2}$$

$$M(t+1) = g[K(t), N(t)] \tag{3}$$

where g is an output function.

An FSM with properties described in the Eqs. (1) and (2) is called a *Moore* Machine and a machine described through the Eqs. (1) and (3) is called the *Mealy* Machine.

The operation of computers is based on the operation of transistors, which depending on the amount of stored charge, the signal can be interpreted as high (1) or low (0), and off (no stored energy).

As the computer works on the interpretation of two electrical impulses can be observed that it is a binary system, therefore, being governed by Boolean algebra.

Boolean algebra is an algebraic structure that defines the arithmetic of logical operators that, being composed of the symbols $S = \{0, 1\}$, constitute a binary system. The concepts of Boolean algebra are also used in electronics since physical circuits are rather designed in abstractions, called logic circuits.

Given a Boolean space, a *variable* is a symbol representing a coordinate in that space. A variable or its negation is called *literal*. The term product is defined as the Boolean product of one or more literals. A minimal term, or *minterm*, is a term product that outputs a value '1'. A circuit with all variables in certain cases can be simplified, eliminating redundancies and having its size reduced. A Boolean function that implies a combination of minterms is called the *implicant* of a function, and an implicant that cannot be reduced, that is, does not imply another function, is called a *prime implicant*. The sum of all implicants and prime implicants of a function is the set of minterms for which the function's result is '1'.

When representing an FSM, usually are used words or letters to refer to states, since the number of flip-flops needed to represent an FSM is calculated similarly to the number of rows in the truth table. When assigning a value to a state, each literal symbolizes the value that will be delivered to a respective flip-flop at a given time. *The joining of the values of each flip-flop is equivalent to the value assigned to a given state of the FSM*.

The values present in the memory element, when combined, represent the current state. The flip-flops are then connected in combinational circuits that change the value contained in the flip-flop at each clock pulse, making the flip-flops start to represent the value assigned to the next state of the machine, going from the current state to the next state.

The combinational circuit responsible for this change of states is the result of simplifying the expressions obtained from the inputs of a given flip-flop and the stimulus that will be given. The circuit receives the flip-flop output value and the machine state stimulus value. The set of output values represent the next state that the state machine will assume.

The state assignment is fundamental when there is the intention to optimize, as it is directly linked to the size of the expression that will make the change between the current state and the next state. Changing the distribution of values drastically affects the size of the expression, which consequently increases the size of the circuit.

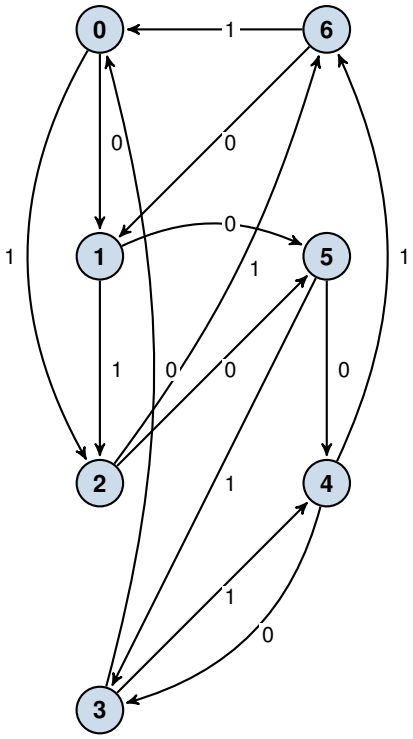
For instance, a 7-state FSM, where the assignment of values to the states is done sequentially, from 0 to 6 with numbers on a binary basis. Table 1 shows the arrangement of assigning values to states. The state transition diagram is depicted in Fig. 1.

The graphical representation of the state transition diagram can also be expressed by the *state transition table* with the transitions as a function of the inputs, as shown in Fig. 1, with the state assignment of the Table 1. In Table 2, where the flip-flops are represented by the variables Q_2 Q_1 Q_0, and d represents the input value that the state will receive. The expressions that generate the value of the next state are given

TABLE 1: FIRST ASSIGNMENT.

State	Assignment
0	000
1	001
2	010
3	011
4	100
5	101
6	110

Fig. 1: Example of a state transition diagram for an FSM.



by Y_2, Y_1, and Y_0.

As the FSM has seven states, it can be represented by three flip-flops, and with the addition of the input, therefore four is the minimal number of values needed to represent the combination of inputs necessary to define the transitions. As a result, the truth table has sixteen rows.

To obtain the expressions, the Karnaugh map simplification method was used, which facilitates the grouping of terms to perform the operations that allow reducing the expression, as illustrated in Fig. 2.

The Karnaugh map is used to simplify and find the respective logical expression for each Y_i. The simplified expression serves as the basis for the construction of the corresponding logical circuit. With this simplification, its obtained:

$$Y_2 = \bar{Q}_2 Q_1 \bar{Q}_0 + Q_1 Q_0 d + \bar{Q}_1 Q_0 \bar{d} + Q_2 \bar{Q}_1 \bar{Q}_0 d$$
$$Y_1 = \bar{Q}_2 \bar{Q}_0 d + Q_2 \bar{Q}_1 \bar{Q}_0 + \bar{Q}_1 d$$
$$Y_0 = \bar{Q}_0 \bar{d} + \bar{Q}_2 \bar{Q}_1 d + Q_2 Q_0 d$$

Performing a new assignment of values to states randomly, instead of doing it sequentially or ordered, can result in several possible combinations, one of which was chosen and represented in Table 3.

TABLE 2: TABLE REFERRING TO THE FIRST ASSIGNMENT OF STATES.

Q_2	Q_1	Q_0	d	Y_2	Y_1	Y_0
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	1	0	1
0	0	1	1	0	1	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	0	0	0
0	1	1	1	1	0	0
1	0	0	0	0	1	1
1	0	0	1	1	1	0
1	0	1	0	1	0	0
1	0	1	1	0	1	1
1	1	0	0	0	0	1
1	1	0	1	0	0	0
1	1	1	0	X	X	X
1	1	1	1	X	X	X

TABLE 3: SECOND ASSIGNMENT OF STATES FOR THE EXAMPLE.

State	Assignment
0	010
1	101
2	000
3	110
4	001
5	011
6	100

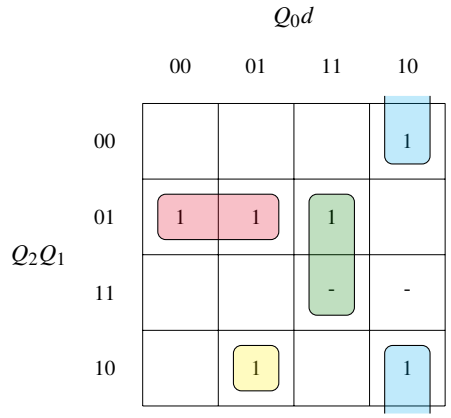
The resulting FSM in the new assignment has the same graph structure and transitions, but with different values assigned to each state. The expression that will be obtained with the simplification makes the resulting circuit different from the previous state assignment. The state transition table referring to the FSM after the new assignments is shown in Table 4.

TABLE 4: TABLE REFERRING TO THE SECOND ASSIGNMENT OF STATES.

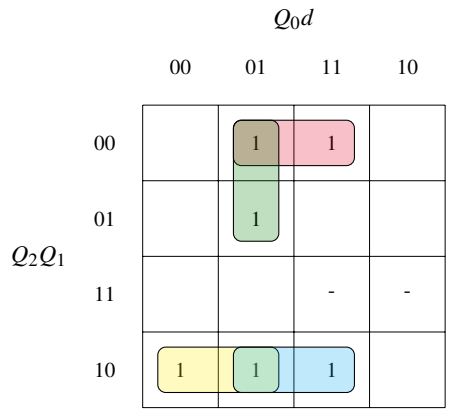
Q_2	Q_1	Q_0	d	Y_2	Y_1	Y_0
0	1	0	0	1	0	1
0	1	0	1	0	0	0
1	0	1	0	0	1	1
1	0	1	1	0	0	0
0	0	0	0	0	1	1
0	0	0	1	1	0	0
1	1	0	0	0	1	0
1	1	0	1	0	0	1
0	0	1	0	1	1	0
0	0	1	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	1	1	0
1	0	0	0	1	0	1
1	0	0	1	0	1	0
1	1	1	0	X	X	X
1	1	1	1	X	X	X

With the new assignment, it is possible to see from the

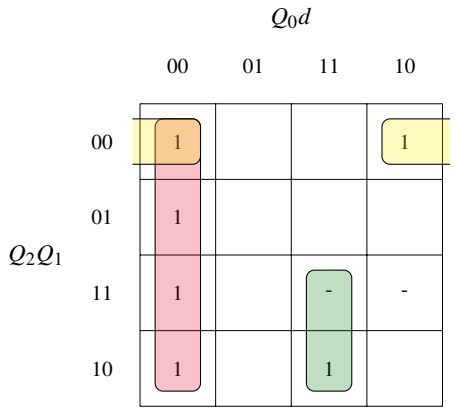
Fig. 2: Karnaugh map to obtain boolean expressions for Y_2, Y_1 e Y_0.



(a) Map of Y_2



(b) Map of Y_1



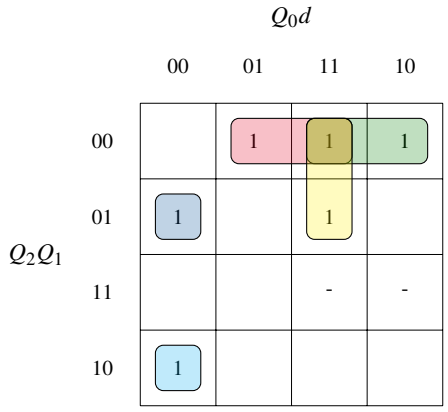
(c) Map of Y_0

resulting Karnaugh maps, shown in Fig. 3, that there will be no minterms of two variables. With the simplification we obtain:

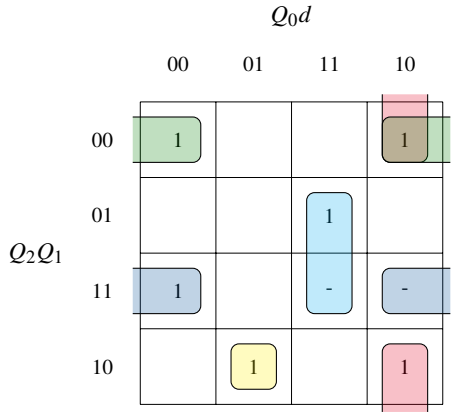
$$Y_2 = \bar{Q}_2Q_1\bar{Q}_0\bar{d} + Q_2\bar{Q}_1\bar{Q}_0\bar{d} + Q_1Q_0d + \bar{Q}_2\bar{Q}_1d + \bar{Q}_2\bar{Q}_1Q_0$$
$$Y_1 = \bar{Q}_2\bar{Q}_1\bar{Q}_0\bar{d} + Q_2Q_1\bar{Q}_0\bar{d} + Q_2\bar{Q}_1\bar{Q}_0d + \bar{Q}_1Q_0\bar{d} + Q_1Q_0d$$
$$Y_0 = \bar{Q}_2\bar{Q}_0\bar{d} + \bar{Q}_1\bar{Q}_0\bar{d} + Q_2Q_1d + Q_1Q_0\bar{d} + Q_2Q_0\bar{d}$$

The result of the simplifications shows that there was a change in the size of expressions. This size difference in an FSM with many states can be drastic, causing a considerable increase in power consumption, physical circuit size, and execution time.

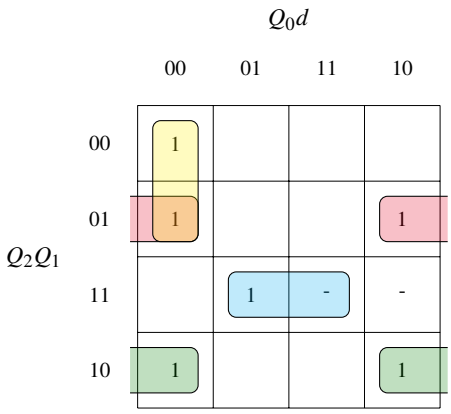
Fig. 3: Karnaugh map to obtain Boolean expressions for Y_2, Y_1 e Y_0 with the second assignment.



(a) Map of Y_2



(b) Map of Y_1



(c) Map of Y_0

b. Simulated Annealing

SA is a technique that simulates the heating and cooling process of materials, allowing the escape of optimal locations, and a better exploration of the search space [13, 14]. When a certain material is heated, there is the excitation of the molecules, and how it cools down to the stability point of the molecules will determine characteristics such as hardness, strength, flexibility, etc.

The way a metal reacts to stress is directly related to how its micro-structure is organized. The capacity of a material deforms under stress is called **Plastic Deformation**[15]. The deformation of a material occurs through rearrangement of the molecules that constitute the crystal grain, a large crystal grain has more molecules to shift during the applying of forces. Since the limit of deformation is related to the size of the grain, a material composed of small crystal grains, when subjected to stress, won't deform as much as a material with large crystal grains.

The annealing process through slow cooling makes the micro-structure of the finished product composed mostly of large crystals, making the material softer, therefore more susceptible to plastic deformation.

In annealing, the metal is heated to a uniform temperature throughout its length, and allowed to cool slowly, gradually, and uniformly. Thus, giving the material a better structuring and organization of the material's molecules, resulting in a more flexible material. The annealing allows the material to be soft, which is better for molding and electrical conductivity.

In the tempering of metal, the material is heated to a temperature close to melting point and cooled abruptly, causing the micro-structure to stabilize with small crystals, with little leeway for deformations, making the material hard.

The computation uses the idea of temperature, which slowly drops to the point of stability. SA as the search and improvement algorithm of Hill Climbing uses this temperature parameter to control when jumps out of the optimal locations occur [16, 17, 13].

The algorithm works as follows: an initial solution is defined, from which the algorithm will start, and a value that will behave like the temperature in thermodynamics, and will decrease in small steps, as in slow cooling. Within each iteration of temperature parameter reduction, another routine of defined size occurs, where another random solution is generated within the search space that will be compared with the initial solution.

This internal routine occurs n times within a temperature adjustment step. In this way, seeking stability of the solution within the temperature range in which the algorithm execution is located. As the temperature parameter decreases, this routine starts to be executed with fewer chances of a new solution being chosen, due to the acceptance criteria [16, 17, 13].

The acceptance of a new solution is based on a thermodynamic model in which starting from a system state i of energy E_i, a new state j of energy E_j is generated based on a permutation.

If the energy difference between current state (i) and new state (j) is greater than zero, the new state is chosen. If the difference is less than zero, the probability that the state j replaces i and becomes the current state is given by[14]:

$$P(\Delta E, T) = e^{\left(-\frac{-(E_j - E_i)}{T}\right)}, \tag{4}$$

where T represents the current temperature.

At the beginning of the algorithm execution, where the temperature is high, there is a greater probability that fewer local solutions will be accepted, promoting the leap to other parts of the search space. But as the temperature drops, this possibility is also present, suitable as chances of jumping to distant points in the search space [16, 17, 13].

III. Methodology for the Experiments

The classic model of the SA algorithm was explained in the previous section, but how certain parts of the work were organized required that the structure of the algorithm be changed to better suit the problem.

How the search space is created for the state assignments for the FSM is random and it is not controlled by any parameter, so there is no way to control whether a solution is in the local neighborhood or a distant point of the space search, or even control the distance of the jump depending on the temperature.

For this reason, the SA applied to the problem only accepts new solutions if the quality is better than the current one, and the probability is favorable. The algorithm works like a Hill Climb where new solutions are accepted based on probability quality. The pseudo algorithm is shown in Algorithm 1.

a. Cost calculation

The cost of a given state assignment is usually calculated by the number of literals in the Boolean expression that represents the FSM. But for comparison reasons, the factor that decides the quality is the area and the type of flip-flops used to build the sequential circuit described by the FSM. The equation is the same as [10]:

$$Area = P \times (2i + 3\log_2 N + o + n_{jk}) \tag{5}$$

where P are the number of product terms, N is the number of states, i is the number of inputs, o is the number of outputs, and n_{jk} is the number of flip-flops JK used in the physical form of the circuit. The JK flip-flops have a more complex structure than the other types of flip-flops, which translates to a larger usage of physical space. The cost calculation penalizes the use of JK flip-flops since it means an increase in the physical space of the resulting circuit. In this paper are used only type D flip-flops, thus we do not utilize the term n_{jk} in the cost calculation.

b. Minimization

One important step in the whole process, is the conversion of the FSMs given by the LGSynth89 benchmark suite [18]. The files are in .KISS2 format are converted to .PLA (Programmable Logic Array) format, then fed to the well-known ESPRESSO [19] Logic Minimizer, a program from the SIS Logic Synthesis System. ESPRESSO is required for calculating the objective function. That is, to obtain the Boolean

Algorithm 1: Pseudo-code of the *Simulated Annealing*

```
Result: Write the results in a plot and a CSV file
Select an initial solution;
Select a starting temperature;
Select a cooling factor;
Select a few loops in each temperature iteration;
while temperature not stabilized do
    for repetition cycles do
        Read the .kiss2 file;
        Convert to a .pla file;
        Get cost of the current solution with ESPRESSO;
        Generates new solution;
        Get cost of the new solution with ESPRESSO;
        Calculates the energy variation using (4);
        if new solution is better than current solution then
            current solution ← new solution;
        end
        else
            if probability allows then
                current solution ← new solution;
            end
        end
        if current solution is better than global solution then
            global solution ← current solution;
        end
    end
    Apply cooling factor;
end
```

expressions that represent the FMS, as explained in Section II.

The ESPRESSO program is a C language implementation of the ESPRESSO algorithm [20, 21], that receives the PLA converted from the .KISS2 file outputs a .PLA format containing the minimized FSM, as well as the number of inputs, number of outputs, and the number of products terms, which represents the state transitions of the FSM, is used in the equation that calculates the cost (Eq. (5)). Conversion is just a form of binary and labeled representation of the FSM. For this reason, the details will be omitted, but details on representation can be obtained in the references.

IV. Results and Comparison

The SA starts with a temperature of 100, in the first half, the cooling factor is 1.2, in the second half the factor changes to 0.8, the same used in[10], slowing down the temperature curve. The initial temperature parameter was chosen based on the tests made, where all the FSM achieve stability by the end of the iterations. The algorithms (both our SA and the methods used in the comparison) were evaluated using the benchmark LGSynth'89 [18]. The benchmark LGSynth'89 is a set of examples with 41 FSM presented in the International Workshop on Logical Synthesis in 1989. The information about the number of states, size of the input, and output of the FSM are shown in Table 5.

To run the tests was used a notebook equipped with Intel i7 6th generation, 16 GB of RAM. The code was written in Python and used the ESPRESSO script, which is written in C language, as a sub-process to minimize the state assignment generated.

The experiment results are shown in Fig. 4. The results are shown normalized. Most cases continue to improve the solution towards the end of 100 iterations. Furthermore, the lines 6 to 21 in Algorithm 1 were repeated 100 times. With a few exceptions, such as `donfile` and `sand`, most of the cases had continued to improve all over the iterations. These cases (`donfile` and `shiftreg`) are small FSM, so there was not much improvement in the solution over the iterations.

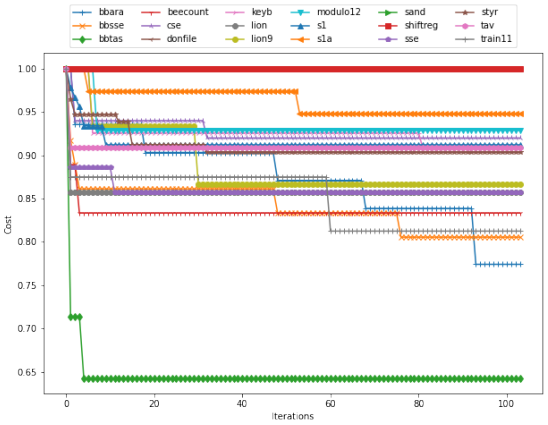


Fig. 4: Improved solution for all cases tested in the experiment compared to all iterations.

Table 6 shows the general results of the experiment carried out and a comparison with the work of [10, 22, 23]. For clarity of the results, all data from the other methods were extracted from the research of [10, 22, 23]. Only the second and tenth columns of Table 6 are the results of this experiment. Columns with the same method and values in parentheses mean that parameters were passed to the method to differentiate the search for the solution. The MUSTANG [2] (MUS for short) was run with `-p` and `-n` options, that correspond to fanout-oriented and fan in-oriented algorithms. The NOVA [24] was run with the `-e ig` option that causes the NOVA to be driven by input constraints and `-e ioh` option that causes it to be driven by both input and out constraints. The JEDI [25] was run with `-e o` option that uses the output dominant algorithm and `-e c` option that uses the coupled dominant algorithm.

The cases in which our results were better than or equal to all the compared methods are highlighted in bold. And it is highlighted in italic, all cases where the results are better than or equal to the methods compared except for the GESA (Guided Evolutionary Simulated Annealing) method. The GESA method is a hybrid algorithm (SA and Genetic Algorithm) available in [10]. The GESA is our focus to compare our experiment.

From these data, it is possible to notice that our results were not far from the compared methods. Compared to GESA, the worst case is `modulo12` where our solution was 59 % worse than GESA. However, the best case was `donfile` where our solution was 10 % better than GESA. On

average, our results were 11 % lower than GESA. It is difficult to measure whether these are acceptable percentages, considering the simplicity of our implementation compared to the GESA hybrid method, it seems reasonable to consider it.

The important point of our results is the processing time. In all cases, better processing times were obtained than the GESA method. For the worst case, the `tav` case, there was an improvement of 63 % of the time. And for the best case, case `s1a`, there was an improvement of 95 %. On average, there was an improvement of 86 % for all cases.

The time improvement was an expected factor, since our algorithm is much simpler than GESA, and it does not work on a certain population as a Genetic Algorithm. One point that could threaten the validity of our experiment is the research age of [10], which is a paper published in 2000. However, it is important to note that GESA was implemented with the C language, whereas in this work the Python language. Python, for being interpreted, is a slower language than the C language. Another point is the hardware that was used. In GESA a SPARCstation 20 station was used. SPARCstation 20 supports up to 4 CPUs, and in the paper, there is no information about which CPU is used. In our experiment, as already mentioned, the Intel i7 6th generation was used. Despite being very different generations, a low-performance personal computer with a high-performance server is being compared.

V. Conclusions

In this research, the problem of finding near-optimal state assignment in a finite state machine has been considered. Moreover, the work shows an evaluation of the solutions provided by a simulated annealing algorithm. Specifically, it has been provided an answer to the following question "how distant are the results with a much simpler and faster metaheuristic with less computational efforts (without a population of solutions)?" and our contribution is a solution to the assignment states in a finite state machine with the near-optimal solution with less computational effort, using Simulated Annealing.

The results have shown that it is possible to have acceptable losses in the quality of the solution with a considerably small amount of processing time, i. e. with less computational effort. For example, for the cases `bbsse` and `cse`, where there was a time gain of 82.93% and 84.52%, respectively, at a cost of loss of solution quality of 2.67% and 2.57%, respectively. Moreover, the case `beyb`, where there has been a time gain of 71.91% with the best of solution with 5.35% of loss quality solution. However, given the gap between the computers used, there must be a fairer comparison of performance. Because it is a much simpler algorithm, it could also be important to carry out an experiment calculating the real computational cost or effort, measured in power or joules per instruction.

References

[1] G. De Micheli, R. Brayton, and A. Sangiovanni-Vincentelli, "Optimal state assignment for finite state machines," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 4, no. 3, pp. 269–285, 1985.

[2] S. Devadas, H.-K. Ma, A. Newton, and A. Sangiovanni-Vincentelli, "Mustang: state assignment of finite state machines targeting multi-level logic implementations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, no. 12, pp. 1290–1300, 1988.

[3] P. Molitor, "A survey on wiring." *Elektronische Informationsverarbeitung und Kybernetik*, vol. 27, pp. 3–19, 01 1991.

[4] R. Dennard, F. Gaensslen, H.-N. Yu, V. Rideout, E. Bassous, and A. LeBlanc, "Design of ion-implanted mosfet's with very small physical dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, 1974.

[5] V. Fabera, V. Janes, and M. Janesova, "Automata construct with genetic algorithm," in *9th EUROMICRO Conference on Digital System Design (DSD'06)*, 2006, pp. 460–463.

[6] N. Niparnan and P. Chongstitvatana, "An improved genetic algorithm for the inference of finite state machine," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 7, 2002, pp. 5 pp. vol.7–.

[7] S. Amellal and B. Kaminska, "Functional synthesis of digital systems with tass," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 5, pp. 537–552, 1994.

[8] B. Mitra, S. Jha, and P. Choudhuri, "A simulated annealing based state assignment approach for control synthesis," in *[1991] Proceedings. Fourth CSI/IEEE International Symposium on VLSI Design*, 1991, pp. 45–50.

[9] G. Hasteer and P. Banerjee, "Simulated annealing based parallel state assignment of finite state machines," in *Proceedings Tenth International Conference on VLSI Design*, 1997, pp. 69–75.

[10] I. Ahmad, F. Ali, and R. Ul-Mustafa, "An integrated state assignment and flip-flop selection technique for fsm synthesis," *Microprocessors and Microsystems*, vol. 24, no. 3, pp. 141–152, 2000.

[11] T. Floyd, *Sistemas digitais: fundamentos e aplicações*. Rio de Janeiro: Bookman Editora, 2009.

[12] H. Taub, *Circuitos Digitais e Microprocessadores*. Rio de Janeiro: Editora McGraw-Hill, 1984.

[13] S. Kirkpatrick, J. C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *SCIENCE*, vol. 220, pp. 671–680, 1983.

[14] E.-G. Talbi, *Metaheuristics: From Design to Implementation*, 06 2009, vol. 74.

[15] N. Hansen and C. Barlow, "17 - plastic deformation of metals and alloys," in *Physical Metallurgy (Fifth Edition)*, fifth edition ed., D. E. Laughlin and K. Hono, Eds. Oxford: Elsevier, 2014, pp. 1681–1764. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780444537706000174

[16] J. Dréo, P. Siarry, A. Pétrowski, and E. Taillard, *Metaheuristics for Hard Optimization*. Springer, 2003.

[17] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics*. Springer, 2019.

[18] S. Yang, "Logic synthesis and optimization benchmarks," Tech. Rep., Dec. 1988. [Online]. Available: https://ddd.fit.cvut.cz/prj/Benchmarks/index.php?page=download

[19] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli, *SIS: A System for Sequential Circuit Synthesis*. Electronics Research Laboratory,Department of Electrical Engineering and Computer Science University of California, Berkeley, CA 94720, 1992.

[20] R. Rudell and A. Sangiovanni-Vincentelli, "Multiple-valued minimization for pla optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 6, no. 5, pp. 727–750, 1987.

[21] ——, "Espresso-mv: algorithms for multivalued logic minimization," *Memorandum No. UCB/ERL M86/65*, 1985.

[22] P. Yip and Y.-H. Pao, "A guided evolutionary computation technique as function optimizer," in *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, 1994, pp. 628–633 vol.2.

TABLE 5: TABLE OF CHARACTERISTICS FOR BENCHMARKS USED

Example	No. of states	No. of inputs	No. of outputs
bbara	10	4	2
bbsse	16	7	7
bbtas	6	2	2
beecount	7	3	4
cse	16	7	7
donfile	24	2	1
keyb	19	7	2
lion	4	2	1
modulo12	12	1	1
s1	20	8	6
s1a	20	8	6
sand	32	11	9
shiftreg	8	1	1
sse	16	7	7
styr	30	9	10
tav	4	4	4
train11	11	2	1

TABLE 6: COMPARISON OF COSTS OBTAINED BY SA.

Case	**SA**	GESA	MUS-P	MUS-N	NOVA(-e ig)	NOVA(-e ioh)	JEDI(-e o)	JEDI(-e c)	Runtime GESA (s)	**Runtime SA (s)**
bbara	*509*	432*	550	572	550	572	616	594	286.92	63.72
bbsse	*990*	900*	1122	1089	990	1089	1122	1089	1091.73	73.82
bbtas	*124*	120*	195	150	180	165	165	195	164.16	48.39
beecount	276	198*	228	228	228	228	209	228	215.12	52.47
cse	*1518*	1480*	1485	1584	1485	1815	1947	1980	1685.96	109.71
donfile	**750**	840	980	1020	960	940	900	620*	1078.40	111.44
keyb	**1408***	1457	3317	1798	1705	3162	1798	1860	1750.82	240.97
lion	*66*	55*	77	77	66	88	88	77	122.12	46.17
modulo12	172	108*	195	195	180	180	165	180	350.64	51.18
s1	**2832**	2849	3108	3552	3219	2775*	3182	2923	1833.00	151.94
s1a	2605	2470	3182	2886	2960	2701	1813*	2479	2638.84	115.16
sand	4600	3901*	5060	5014	4692	4554	4876	4830	2256.48	277.05
shiftreg	**48***	48	48	72	96	48	132	96	198.45	47.65
sse	*974*	875*	1122	1089	990	1089	1188	1122	1060.20	73.66
styr	*4400*	3854*	5117	4945	4429	4558	4816	4644	1845.40	385.75
tav	*180*	162*	198	198	198	198	198	198	139.22	50.68
train11	200	147*	238	238	204	187	221	187	495.59	51.82

[1]Values in bold mean improvement in the solution. Values in italic mean a deterioration of the solution.
The value marked with a '*' is the best solution for given benchmark

[23] ——, "Combinatorial optimization with use of guided evolutionary simulated annealing," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 290–295, 1995.

[24] T. Villa and A. Sangiovanni-Vincentelli, "Nova: state assignment of finite state machines for optimal two-level logic implementation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 9, no. 9, pp. 905–924, 1990.

[25] F. Buijs and T. Lengauer, "Synthesis of multi-level logic with one symbolic input," in *Proceedings of the Conference on European Design Automation*, ser. EURO-DAC '91. Washington, DC, USA: IEEE Computer Society Press, 1991, p. 60–64.