XVIII Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 2018)

CAEPIA 6:
MULTIMEDIA E IMÁGENES

A software tool for categorizing violin student renditions by comparison*

*Note: The full contents of this paper have been published in the volume *Lecture Notes in Artificial Intelligence 11160* (LNAI 11160)

Miguel Delgado, Waldo Fajardo, Miguel Molina-Solana
Department of Computer Science and Artificial Intelligence
University of Granada
Granada, Spain
mdelgado@decsai.ugr.es, {aragorn, miguelmolina}@ugr.es

***Abstract*—Music Education is a challenging domain for Artificial Intelligence because of the inherent complexity of music, an activity with no clear goals and no comprehensive set of well-defined methods for success. It involves complementary aspects from other disciplines such as psychology and acoustics; and it also requires creativity and eventually, some manual abilities. In this paper, we present an application of machine learning to the learning of music performance. Our devised system is able to discover the similarities and differences between a given performance and those from other musicians. Such a system would be of great value to music students when learning how to perform a certain piece of music.**

***Index Terms*—**

Towards an Automated Composer of Popular Spanish Songs: Integrating a Music Generator and a Song Lyrics Generator

María Navarro-Cáceres
Computer Sciences Department
University of Salamanca
Salamanca, Spain
maria90@usal.es

Hugo Gonçalo Oliveira
CISUC, Department of Informatics Engineering
University of Coimbra
Coimbra, Portugal
hroliv@dei.uc.pt

Juan M. Corchado
Computer Sciences Department
University of Salamanca
Salamanca, Spain
corchado@usal.es

***Abstract*—We describe the integration of two creative systems in what can be seen as an automated composer for Spanish popular music. We first present a system that generates melodies with a Markov Model learned from a corpus of Spanish popular music. Then, given the importance of the lyrics in this context, the latter was integrated with an existing lyrics generation system, adapted to suit this purpose. Several experiments were carried out to evaluate the quality of the results. Overall, melodies transmit a feeling of Spanish popular music, while the text of the lyrics is acceptable for a first approach.**

***Index Terms*—Computational Creativity, Music Generation, Lyrics Generation**

I. Introduction

We can say that, today, Computational Creativity [1] is an established subfield of Artificial Intelligence. One that has attracted researchers with multiple origins and motivations, and where a variety of intelligent systems has been developed for the generation of a wide range of artifacts with a creative value, such as music [2], [3] or poetry [4], [5].

However, few creative systems have tackled both the previous at the same time, towards the production of music with lyrics. The main contributions of this work is the development of an automated composer, in this case focused on popular Spanish songs. It is based on the integration of two systems: ETHNO-MUSIC, a new system for generating popular melodies; and Tra-la-Lyrics [6], longer-established, which generates lyrics for a given rhythm. This is in line with recent collaborations between different creative systems (e.g., [7], [8]) and shows that, with the current landscape of creative systems, it is not always necessary to develop new systems from scratch.

On the specifics of the new system, most initiatives on music generation have focused on music with a rather tonal character and, to date, there has been no study addressing the generation of Spanish popular music. This genre differs from classical music in many aspects, including the sonority (meaning the tonality of the music), the sound disposition or the rhythmic formulas used. The automatic generation of this kind of music also depends on multiple factors that are intrinsically connected, such as the representation of the tonality, the melodies and the rhythm. Moreover, unlike classical music, Spanish popular music is always linked to a functionality. In this sense, lyrics are essential for identifying the purpose for which the music was conceived, i.e. if it is a work song or a love song.

Drawing on this phenomenon, new melodies are generated, based on original Spanish popular songs in multiple sources of popular music. Musical excerpts were then analyzed and their relevant features were encoded and stored to be used as the training corpus for music. Among the different learning models that have been applied to generate music from a corpus, Markov Models (MM) were selected due to their successful application in related works [2], [3], [9].

Generation of lyrics starts once the melody is ready. Simple heuristics are applied for splitting the melody into parts, and then lines of text are generated for each part, while trying to maximize two main constraints: (i) one syllable per note; (ii) stressed syllables matching strong beats of the melody. The current version of Tra-la-Lyrics is built on top of PoeTryMe [5], a platform for poetry generation in different languages. In this case, generation is based on the Spanish instantiation, though with an augmented semantic network and new line templates, acquired automatically from songs in Spanish. In order to set the generation domain and establish a connection between lyrics and the subjects typically addressed in Spanish popular songs, seed words were carefully selected.

In the end, the resulting process is analogous to having two different people composing a song: one that composes the melody and another the lyrics. In this case, the melody is composed first, unlike other creative systems,

which generate music for given lyrics (e.g. [10], [11]). Given that the composition of a song by humans may follow a different order – starting with the lyrics vs starting with the melody –, we believe that it makes sense to tackle this challenge both ways.

Out of the compositions resulting from our integration, some were selected for human validation. Overall, the most positive aspects were that melodies were pleasant and had a sound and rhythm that gave a feeling of the Spanish popular songs, and that the text of the lyrics was within the target domains. Less positive aspects, but still average, were the rhythm and meaning of the lyrics. In the remainder of this paper, the integration effort is described with more detail and illustrative examples of generated songs are presented and discussed, together with the validation results.

II. Integration

The generation of popular Spanish songs results from the integration of two creative systems: ETHNO-MUSIC, in charge of generating melodies, and Tra-la-Lyrics [6], in charge of lyrics generation. Figure 1 gives an overview of the resulting generation flow.

ETHNO-MUSIC is provided with a memory to store beforehand different melodies. It takes the melodies stored and trains a model for generating a new composition. For the purposes of this paper, Markov Models (MM) were used as the learning algorithm. Once the melody is available, Tra-la-Lyrics is used for generating lyrics that suit the melody.

A. Music Generation

In the current work, music is generated based on the features of Spanish popular music, which include the rhythm or the sonority.

1) Music Retrieval: To generate melodies following the popular song standards, MMs are applied. They need a set of musical sources for the learning process, which should reflect the most common sonorities of the Spanish popular music. Therefore, the melodies are extracted from specialized songbooks, recordings and digital scores.

According to ethnomusicologists [12], three main factors should be considered in Spanish popular music: melodic behavior, rhythm and style. Regarding the melody, the modal music in Spain can follow seven diatonic modes (Jonic, Dorian, Phrigian, Lidian, Mixolidian, Eolian and Locrian). The melodic behavior consists of continuous chromatizations and instabilities. In addition, melodies do not follow any predetermined scheme in terms of tonic and dominant functionalities, as they are based on modal paradigms. Popular music also contains a very uniform beat due to the syllabic text used in these songs, because the prose in this type of repertoire has a regular rhythmic nature, thus totally avoiding a prosodic rhythm. Finally, one of the key aspects of popular music is their style, based on functionality, namely the context in which the music has been conceived. In this sense, we can classify music in different genres, each one representing their own musical features, such as sonority, lyrics and rhythm. Consequently, popular music could be classified accordingly as work songs, love songs, lullabies, wedding songs, sacred songs, dance music, and others.

Despite the general features shared by most popular songs, there was a series of musical parameters (i.e. rhythms or sonorities) spread throughout the Iberian peninsula that, over time, became different in each region or even disappeared in some. According to some authors [13], in Spanish popular music, the work, love songs and lullabies share features related to the key signature, rhythmic patterns and general sonority or tonality, which made them very interesting to use as a corpus in the development of the learning model. Therefore, in order to train the MM, melodies related to these genres that shared some common features were collected.

2) Encoding Melodies: To represent the information needed for the training of the MM, sources were digitalized. To identify popular music, we did not only analyze the particular duration or pitch, but also the duration of the musical phrases, the degrees in which the melody reposes (notes with a long duration), and the particular cadences. Drawing on these properties and also inspired by a concept of viewpoints [14], Table I shows the features encoded for training the MM.

TABLE I
Musical features encoded in the songs and used in the training of the MM.

Feature	Description
Pitch	MIDI number that corresponds to the musical note
Duration	Number that represents the rhythmic formula or one note. Each number can represent a whole, a half, a quarter, etc.
Degree	Number from 1 to 7 that represents the degree of each note according to the scale — 1 means the first degree, 7 means the last note in the scale
First in bar	Boolean number, where 1 represents the first note in a bar, otherwise it is 0
Time Signature	Represented by two numbers, one for the numerator and another for the denominator
Musical phrase	Represents the position of a note in a musical phrase, and can take three possible values: 1 if the note is the first in the phrase, -1 if it is the last one, and 0 otherwise

Currently, many songs are already encoded as MIDI (Musical Instrument Digital Interface) [15]. This is due to the availability of this format throughout the Web, the low difficulty in creating such files based on digital scores, and their structure, which allows a relatively easy access to musical features. Although XML or other encodings like MusicXML or MEI are also available and also encode musical features, we selected MIDI because

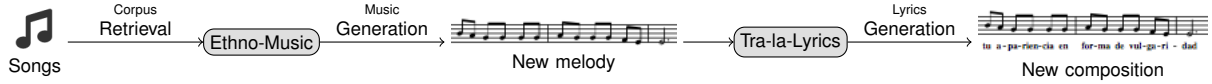


Fig. 1. Overview of music generation flow.

it is the format which most appears in the web, and which can be listened in almost every device. MIDI files include instructions that allow the reconstruction of the song by a sequencer and a synthesizer that works with MIDI specifications.

MIDIs provide information on musical events like pitch, duration and time signature, extracted for the purposes of this work. Yet, other, such as phrase construction, main sonority, melodic degree (position that a note takes in the scale) and bars, have to be manually added. This information has been encoded by following the codification proposed by [16], which use numbers to describe some musical parameters. These data are finally stored in a CSV file.

3) Generating the Melody: Once the files are available, the necessary information about the music is extracted. The data previously analyzed and incorporated in the files are used as to train the learning model. First, a model is learned for representing the data, which may be used for generating music. Among the possible approaches, MMs are a widely used tool to model the temporal properties of various phenomena, from the structure of a text to economic fluctuations, as well as content generation applications, such as text or music generation. MMs are a specific case of Bayesaian Networks, in which the probability of an event occurring (in this case, a note or silence with a certain duration) depends only on the N-previous events. In this particular case, *N* has been established empirically to 4.

The music composition system modeled tries to create only one melody with a specific rhythm. Therefore, a MM has been learned to generate the melody and the rhythm at the same time. Once trained, each next transition (i.e, each note) is iteratively calculated by applying the probabilities of the MM according to the previous stages. Finally, each note generated by the learning model is encoded in MIDI, to be used in a standard synthesizer.

B. Lyrics Generation

Tra-la-Lyrics [6] generates text based on rhythm, and is currently built on top of PoeTryMe [5], a versatile poetry generation platform. Tra-la-Lyrics can thus be seen as instantiation of the latter. It that shares most of PoeTryMe constituents, including the line generation module, which produces semantically coherent lines, with the help of a semantic network, and a grammar with line templates. Therefore, Tra-la-Lyricis follows a generate-and-test strategy to generate lyrics with the music already created.

Yet, in addition, Tra-la-Lyrics has a parts analyzer, for splitting the melody in parts; a rhythm analyzer, for extracting the rhythm of the melody; and a generation strategy that considers the position of the stressed syllables. Briefly, lyrics are produced after splitting the melody in smaller parts and getting a representation of the rhythm as a sequence of strong and weak beats. The generation strategy selects the best lines for each part, out of those generated by the line generation module.

1) Parts Analyzer: In PoeTryMe, a generation strategy selects and organizes lines, produced by the line generator, according to a given form. The end of a line is a suitable place for rhymes. For poems, the form is given by the number of stanzas, lines and target lengths. Given a melody, the limits of lines have to be identified automatically. We thus set simple heuristics for splitting melodies into parts, based on the lengths of the longest pause (*LP*) and longest note (*LN*): the melody is split after pauses that are $LP/3$-long or longer and notes that are $LN/2$-long or longer, if resulting parts have at least *minP* notes. Parts with more than *maxP* notes go through the same splitting process until they have less than *maxP* notes. The heuristics are similar to the LBDM algorithm for music segmentation [17], but only applied to rhythm, rather than pitch segmentation.

2) Rhythm Analyzer: The rhythm analyzer is the same as in the original version of Tra-la-Lyrics [18]. It is based on the dot system [19], which sets the metrical accents of each beat inside a bar, and thus the strengths of each note, according to their position. The first beat of each bar is always the strongest. Strengths are then distributed according to the beats (stronger) and downbeats (weaker), depending on the time signature. Tra-la-Lyrics only considers those that are substantially stronger, when compared to the remaining, as actual strong beats. For instance, in a 4/4, those will be the first and the third crotchets, and in a 3/4, only the first crotchet is used as the only strong beat.

3) Generation Strategy: Similarly to other instantiations of PoeTryMe, lines are selected with a generate-and-test approach. Briefly, for each line in the poem structure, textual fragments are retrieved from the line generation module. This strategy is responsible for selecting the fittest fragments for each line, from a maximum of *n* retrieved fragments per line. The fitness function is based on features that are relevant to the rhythm. It has penalties for: the difference between the number of syllables in a textual line and the number of notes in the musical part (α); unstressed syllables in strong beats (β); stressed syllables in weak beats (γ); and words interrupted by a pause (δ). In order to increase

the chance of rhymes, there is a bonus for each pair of nearby lines with the same termination (ϵ), unless they end with the same word, which results in another penalty (ζ).

4) Line Generation: PoeTryMe has line generation modules for producing text fragments in Portuguese, English and Spanish, with a semantic network and a grammar with templates, tightly connected with the relation types in the network. Towards generation within a semantic domain, the network is first constrained by a set of seed words. Line generation involves selecting a relation instance (e.g. computer usedFor work; guitar usedFor music; hat usedFor shade), and using its arguments for filling the placeholders of a suitable rule (e.g. usedFor $\rightarrow$ *be my [arg1], i'll be your [arg2]*).

Given the kind of text to generate, it made sense to use the Spanish instantiation, though with some additions that enable the generation of more varied lines. In the Spanish adaptation, semantic relations covered mainly synonyms and hypernyms, while lines of the generation grammar had been extracted from an anthology of about 400 Spanish poems. For this work, the semantic network was enriched with relations between two Spanish words in the most recent version of ConceptNet [20]. Moreover, for the creation of the grammar, a set of about 9,000 Spanish song lyrics, retrieved from the MusixMatch database[1] was also exploited. To complete the adaptation, lyrics had to be generated with seed words related to concepts typically invoked in Spanish popular lyrics.

III. Results

The integration of ETHNO-MUSIC and Tra-la-lyrics results in a new system for the generation of songs, covering melody and lyrics. Here, we describe the settings used towards the generation of several Spanish popular songs and discusses some generated examples.

A. Music Generation Settings

For the generation of the music, 102 popular songs were selected, namely 68 work songs and 34 love songs, all using similar rhythm patterns, with a time signature of 3/4. Each song consisted of 3 or 4 musical phrases with similar length, with the same sonority, the Phrigian mode with possible modifications in its evolution to E minor.

Melodies were encoded as described earlier and saved in a CSV file. Their features were the corpus for training the MM. The number of states that it can remember for the future generation of the melody was empirically set to 4. During the generation process, each iteration of the MM consists of adding a new note to the melody. The MM is iterated until four musical phrases have been created.

For the experiments, a total of 40 melodies were generated with ETHNO-MUSIC, out of which five we were happy with were selected for lyrics generation by Tra-la-lyrics.

[1] https://www.musixmatch.com

B. Lyrics Generation Settings

In order to split the melody in parts, we empirically set *minP* = 4 and *maxP* = 18. For each of the five melodies, lyrics were generated with groups of seed words that set two generation domains, common in Spanish popular songs:

- Work in the fields: *trabajo, siega, tierra, sembrar, semillas, trigo, cereales, campo, sol, paja, cosecha, cosechar* (in English, work, harvest, land, sow, seeds, wheat, cereals, field, sun, straw, harvest, harvest)
- Love: *amor, novia, moza, mozo, bella, belleza, feliz, alcoba, morena, guapa, sonrisa, ojos, bonito, bonita* (in English, love, girlfriend, girl, lad, beautiful, beauty, happy, bedroom, brunette, pretty, smile, eyes, pretty)

For each melody-domain pair, 10 lyrics were generated. Though, out of each 10, only the one with the best rhythm-based score, computed automatically, was selected for human validation, which makes a total of 10 songs.

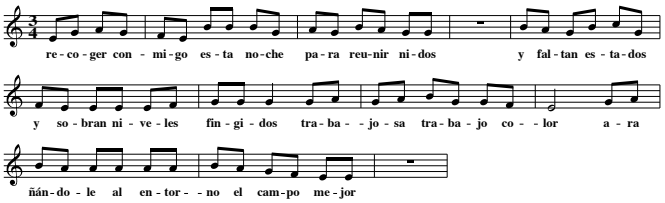
For each line, we empirically set the number of generations $n = 1,750$. In order to match the target rhythm, the parameters of the fitness function were: $\alpha = 1$, $\beta = 0.5$, $\gamma = 0.1$, $\delta = 0.3$, $\epsilon = 2$, $\zeta = 1$.

C. Examples

Figures 2 and 3 show examples of two generated songs, selected from the validation sample. For each example, we present the score, the Spanish lyrics assigned to the corresponding notes, and a rough English translation of their text. All the melodies generated share representative features of popular music, such as the constant repetition of pitches and the limited tessitura. Additionally, to some extent, lyrics match the rhythm and use a varied range of words related to the selected topics.

In the melody of figure 2, the limited tessitura is clear as the notes go from E to B. The rhythmic formulas are very repetitive and centered on the syllabic text that accompanies the melody, instead of showing complex rhythm figures. The melody rests on the E note and the third degree, which is typical of modal music, unlike tonal, where rests are preferred in the fifth or tonic degrees. Lyrics for this song were generated with the work-related seeds, which is clear by the presence of words like *trabajo* (work) or *campo* (field), from the seed set, and also other related words, such as *recoger* (to collect, related to harvest). On the rhythm, a minority of unstressed syllables is on strong beats, namely the first syllable of *fin-gi-dos* and the last of *en-tor-no*, but this is somehow compensated by the presence of two rhymes, *nidos/fingidos* and *color/mejor*.

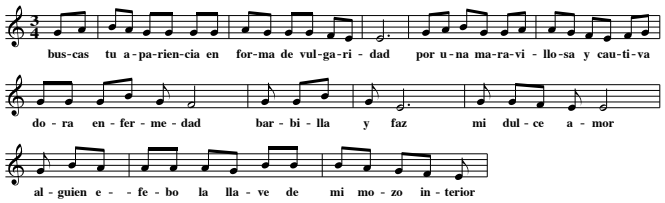
In the melody of figure 3, besides the repetitive rhythm, there are musical rests, another desired feature. The use of short musical phrases is very typical in Spanish popular songs because they are thought for short lyrical phrases that people can remember. Lyrics for this song were generated with the love-related seeds, which results in the presence of words like *amor* and *mozo*, both



Take with me this night to join nests
We run out of terraces and exceed pretended levels
Hard color working scratching the better field

Fig. 2. Song in Phrigian mode, with Spanish lyrics on the domain 'trabajo' (work), including rough English translation.

in the seed set, and indirectly related words such as *maravillosa* (wonderful, related to pretty) or *barbilla* (chin, a part of the face, as the eyes). On the rhythm, we identify two unstressed syllables in strong beats, namely the first syllable of *bar-bi-lla* and the first of *al-guien*. This is again compensated by the presence of two rhymes, *vulgaridad/enfermedad*, *amor/interior*.



You look your appearance in the shape of vulgarity
because of a wonderful and irresistible disease
Your chin and face, my sweet love
somebody stole the key of my heart

Fig. 3. Song in Phrigian mode, with Spanish lyrics on the domain 'amor' (love), including rough English translation.

IV. Validation

Once designed and implemented, we got insights on the validity of the new automated composer. Due to the inherent subjectivity of human listeners, this kind of evaluation remains a challenge for music composed automatically [21]. The same happens for creative text. Following other examples of subjective evaluation by a group of human listeners (e.g. [22]–[24]), who perform listening tests , we follow a similar approach for evaluating the musical results and the lyrics of the system.

More precisely, ten generated songs (five melodies with two lyrics for each) played with a MIDI synthesizer and shown in a score with the notes and the lyrics, were presented to 7 users, experts in popular music. For each song, we asked their opinion on the musical aspects, the text of the lyrics and on their connection, with the following questions, to be rated with a 5-point Likert-scale, between 1 (poorly) and 5 (perfectly well):

1) Melody: How pleasant is the melody?
2) Sound: How well does the melody, in some way, give a feeling of the popular style of the songs?
3) Rhythm of the Melody: How well does the rhythm suit the popular music style?
4) Rhythm of the Lyrics: How well does the text suit the rhythm of the original melody?
5) Subject: How is the text related to any of the following topics: work, love?
6) Meaning: How much sense does the text of the lyrics make? Is it possible to, somehow, interpret it?
7) Overall quality: In general, what is the quality of the melody plus lyrics?

Some of the questions involve an appreciation of music and text as a whole (Rhythm of the lyrics, Overall) and others are more focused on only one of the previous. Yet, although each of the systems had been somehow validated on their own, having both music and lyrics may influence the way that the results are perceived by humans. Plus, this instantiation of PoeTryMe/Tra-la-Lyrics has significant differences from previous (e.g., music is automatically split in parts, lyrics are generated in Spanish, text of Spanish lyrics was exploited, as well as ConceptNet for additional knowledge).

We expect the system to reflect the perceptual quality of the melodies according to the popular songs style. Likewise, we wanted to verify the quality of the lyrics according to the positive ratings given by the listeners. Table II presents the mode (M_o) and median (M_d) ratings for each assessed item.

TABLE II
Overall validation results for the 10 assessed songs.

Item	Rating 1	2	3	4	5	M_o	M_d
Melody	0	3	21	36	10	4	4
Sound	1	4	22	39	4	4	4
Rhythm (Melody)	2	9	23	34	2	4	4
Rhythm (Lyrics)	3	13	27	25	1	3	3
Subject	2	12	23	25	8	4	3
Text Meaning	7	17	28	15	3	3	3
Overall Quality	3	5	36	24	2	3	3

For every assessed item, the majority of the ratings falls in 3 or 4. Yet, the proportion of 4s is higher for the melody-related aspects than for the text-related, where the proportion of 3s is higher. This suggests that the melody is pleasant, and the sound follows quite well the standards of Spanish popular music, as does the rhythm. The worst scoring item is text meaning, which shows that generating text on a topic is only halfway for generating a meaningful test. On the other hand, the quality of the lyrics is lower, but still average. The overall quality of music and lyrics is also average.

V. Conclusion

We described the effort and results of integrating a music and a lyrics generator towards the development of an intelligent system that creates popular Spanish songs with lyrics, automatically. The music generator is

based on a MM, trained on features extracted from a corpus of Spanish popular music. The lyrics generator produces textual fragments that match a given rhythm, with syntactic coherence handled by a grammar and semantics controlled by the combination of the previous grammar with a semantic network.

We see this as a successful effort, because the resulting system generates melodies that follow the Spanish songs standards, with Spanish lyrics on topics typically covered by the popular songs. This is confirmed by a human evaluation which also revealed that the less positive aspects, though still average, are the rhythm of the lyrics and, especially, their meaning.

Besides the success of integrating two different systems developed independently, given that this was our first approach to our goal, we can say that we are happy with our results. Yet, during this work, we identified limitations, some with the help of the human validation, to be addressed in the future, for a new version of the composer.

Regarding the lyrics, despite the utilization of a mature system for generation, alternative strategies could be tested for matching the rhythm. Although the instantiations of PoeTryMe often use a generate-and-test approach, an evolutionary strategy, available in the same platform, could make more sense for lyrics, where the fitness function has more parameters. Generating meaningful lyrics is always a challenging task and we are aware of the limitations of PoeTryMe on this topic. We will investigate this issue further. Yet, natural improvements would be using a larger corpus for extracting the grammars or, for the specific case of this work, use a corpus of lyrics of Spanish popular songs, which we still could not find. Given the importance of lyrics in popular music for ethnomusicologist studies, a deeper analysis of popular songs and their lyrics should be addressed towards the generation of better lyrics of this kind.

An improvement on the integration process would be to have the music generation system providing the division of musical parts directly to the lyrics generation system, together with the melody. This would possibly avoid the heuristics currently applied for this purpose. In order to improve the automatic generation of music, more features of melodies could be used, including a more general view of the composition. It would also be interesting to test singing voice synthesis software for singing the generated songs, which could possibly make the validation clearer.

Acknowledgements

This work was supported by the Spanish Ministry of Economy and FEDER funds. Project. SURF: Intelligent System for integrated and sustainable management of urban fleets TIN2015-65515-C4-3-R.

References

[1] S. Colton and G. A. Wiggins, “Computational creativity: The final frontier?” in *Frontiers in Artificial Intelligence and Applications*, vol. 242, 2012, pp. 21–26.

[2] F. Pachet and P. Roy, “Markov constraints: steerable generation of Markov sequences,” *Constraints*, vol. 16, no. 2, pp. 148–172, 2011.

[3] A. Papadopoulos, P. Roy, and F. Pachet, “Assisted Lead Sheet Composition Using FlowComposer,” in *International Conference on Principles and Practice of Constraint Programming*. Springer, 2016, pp. 769–785.

[4] P. Gervás, “WASP: Evaluation of different strategies for the automatic generation of spanish verse,” in *Proceedings of AISB’00 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science*, Birmingham, UK, 2000, pp. 93–100.

[5] H. Gonçalo Oliveira, R. Hervás, A. Díaz, and P. Gervás, “Multilingual extension and evaluation of a poetry generator,” *Natural Language Engineering*, vol. 23, no. 6, pp. 929–967, 2017. [Online]. Available: https://doi.org/10.1017/S1351324917000171

[6] H. Gonçalo Oliveira, “Tra-la-lyrics 2.0: Automatic generation of song lyrics on a semantic domain,” *Journal of Artificial General Intelligence*, vol. 6, no. 1, pp. 87–110, December 2015, special Issue: Computational Creativity, Concept Invention, and General Intelligence.

[7] M. Znidarsic, A. Cardoso, P. Gervás, P. Martins, R. Hervás, A. O. Alves, H. Gonçalo Oliveira, P. Xiao, S. Linkola, H. Toivonen, J. Kranjc, and N. Lavrac, “Computational creativity infrastructure for online software composition: A conceptual blending use case,” in *Proceedings of the 7th International Conference on Computational Creativity*, ser. ICCC 2016, 2016, pp. 371–379.

[8] E. Concepción, P. Gervás, and G. Méndez, “Afanasyev: A collaborative architectural model for automatic story generation,” in *Proceedings of the 5th AISB Symposium on Computational Creativity*, University of Liverpool, UK, 2018.

[9] R. P. Whorley and D. Conklin, “Music generation from statistical models of harmony,” *Journal of New Music Research*, vol. 45, no. 2, pp. 160–183, 2016.

[10] J. M. Toivanen, H. Toivonen, and A. Valitutti, “Automatical composition of lyrical songs,” in *Proceedings of 4th International Conference on Computational Creativity*, ser. ICCC 2013. Sydney, Australia: The University of Sydney, June 2013, pp. 87–91. [Online]. Available: http://www.computationalcreativity.net/iccc2013/download/iccc2013-toivanen-toivonen-valitutti.pdf

[11] M. Ackerman and D. Loker, “Algorithmic songwriting with ALYSIA,” in *Computational Intelligence in Music, Sound, Art and Design - 6th International Conference, EvoMUSART 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings*, 2017, pp. 1–16. [Online]. Available: https://doi.org/10.1007/978-3-319-55750-2_1

[12] M. Manzano Alonso, “Cancionero popular de burgos,” *Dip. de Burgos*, 2001.

[13] K. Schindler, *Música y poesía popular de España y Portugal*. Centro de Cultura Tradicional, 1991.

[14] R. P. Whorley, G. A. Wiggins, C. Rhodes, and M. T. Pearce, “Multiple viewpoint systems: Time complexity and the construction of domains for complex musical viewpoints in the harmonization problem,” *Journal of New Music Research*, vol. 42, no. 3, pp. 237–266, 2013.

[15] S. Jungleib, *General Midi*. AR Editions, Inc., 1996.

[16] D. Conklin and C. Anagnostopoulou, “Representation and discovery of multiple viewpoint patterns.” in *ICMC*. Citeseer, 2001.

[17] E. Cambouropoulos, “The local boundary detection model (lbdm) and its application in the study of expressive timing.” in *ICMC*, 2001.

[18] H. R. Gonçalo Oliveira, F. A. Cardoso, and F. C. Pereira, “Tra-la-Lyrics: an approach to generate text based on rhythm,” in *Proceedings of the 4th International Joint Workshop on Computational Creativity*. London, UK: IJWCC 2007, June 2007, pp. 47–55.

[19] F. Lerdahl and R. Jackendoff, *A generative theory of tonal music*. Cambridge. MA: The MIT Press, 1983.

[20] R. Speer, J. Chin, and C. Havasi, “Conceptnet 5.5: An open multilingual graph of general knowledge,” in *Proceedings of 31st AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, 2017, pp. 4444–4451.

[21] M. Pearce, D. Meredith, and G. Wiggins, “Motivations and methodologies for automation of the compositional process,” *Musicae Scientiae*, vol. 6, no. 2, pp. 119–147, 2002.
[22] M. Delgado, W. Fajardo, and M. Molina-Solana, “Inmamusys: Intelligent multiagent music system,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 4574–4580, 2009.
[23] M. T. Pearce and G. A. Wiggins, “Evaluating cognitive models of musical composition,” in *Proceedings of the 4th international joint workshop on computational creativity*. Goldsmiths, University of London, 2007, pp. 73–80.
[24] T. Collins, R. Laney, A. Willis, and P. H. Garthwaite, “Developing and evaluating computational models of musical style,” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 30, no. 1, pp. 16–43, 2016.

Goal-Reasoning in StarCraft: Brood War through Multilevel Planning

Moisés Martínez
King's College London
London, United Kingdom
moises.martinez@kcl.ac.uk

Nerea Luis
Universidad Carlos III de Madrid
Leganes, Spain
nluis@inf.uc3m.es

***Abstract*—Real-Time Strategy video-games (RTSVGs) are challenging for most deliberative approaches, such as Automated Planning. This is due to (i) the dynamic changes of the environment; (ii) and the wide variety of potential actions that can be performed over the environment. The aim is "*to win the match*". Besides, RTSVGs presents an additional challenge: managing goals during the game is extremely hard. They change as the game state evolves either because of actions performed by the different agents (player and opponents), by new available information or by unexpected changes of the environment. Thus, generating a detailed sequence of actions –plan– to win the match is not effective in the long term.**

In this paper, we propose an autonomous approach based on two levels of declarative Automated Planning. They are included inside a planning and execution architecture. The high-level, macromanagement, searches and suggests a set of soft-goals according to the current state and the available features of the agent. The low level, micromanagement, generates short plans of actions to reach the soft-goals generated by the high level. We claim that the ability of self-generating goals improves the plan generation and execution performance in a dynamic environment. Finally, we present a preliminary empirical evaluation of this approach tested on StarCraft: Brood War.

***Index Terms*—Automated Planning, Goal-Reasoning, Planning and Execution, Cognitive Systems, Video-Games**

I. Introduction

In recent years, video-games have gotten attraction by the Artificial Intelligence (AI) research community, as they offer challenging environments to test different AI techniques. Specially, StarCraft, which has become popular due to competitions such as SCAAIT [1], AIIDE[2] and the CIG Competition[3]. These competitions provide APIs and tools to easily deploy different AI approaches into StarCraft.

RTSVGs are considered a sub-genre of strategy games e.g. chess, backgammon, go, monopoly, etc. in which players develop a settlement based into two principles: (i) an economy i.e. exploitation of natural resources to build throughout the environment; (ii) and a military power i.e training different units and researching technologies which offer advantages over the enemy. These features hinder the application of traditional AI approaches i.e. the game complexity is extremely large in terms of the size of the state space and the search space and in the number of actions that can be executed on each decision step. For instance, a StarCraft map is defined as a rectangular grid where each tile is measured in building tiles. The smallest map has 64 x 64 building tiles. Thus, if a player builds the barracks (3x4), there exist 4096 tiles where they can be built in the worst case. If the player has four workers, there are 16384 instantiated actions available. Besides, a StarCraft match is executed at 24 frames per second, which means that the game state changes every 42 ms. The environment is non fully-observable, players just know the part of the environment that has been previously explored or built. In addition, the StarCraft environment is continuous, which means that players' turns do not explicitly exist. Actions are executed when requested by the players. Some actions can be executed in parallel. Besides, actions in this kind of games are durative. Thus, there is a small delay after the player chooses the action until it is executed on the game.

In order to win a match in a RTSVG like StarCraft, we must generate and execute a sequence of repetitive actions (plan) to defeat the opponent. But, to achieve this final goal, we must achieve first a huge number of small goals which appear during the game based on our previous actions and the new information discovered. Automated Planning (AP) is able to generate a sequence of actions that solves a specific problem like this one. Goals are usually given as input or as a part of the problem representation. However, in complex environments such as StarCraft, it is not possible to initially define a set of goals due to two important reasons: (i) the dynamism of the environment; (ii) the capabilities of the player, improved or penalized according to the actions executed over the environment. Nonetheless, several approaches based on AP have been successfully used before to solve complex problems with partial information such as the planning Mars exploration missions [1], managing fire extinctions [5], controlling underwater vehicles [20], intermodal transportation problems [8] and controlling quadcopters [3]. The key was to include some specific knowledge that simplified the deliberative act of reasoning.

In this paper, we propose a two level planning architecture. The high level consists on a goal-reasoning process, which generates goals based on the available information from the match e.g. player, environment, opponent etc. The low level consists on a deliberative process, which generates small plans

[1]http://sscaitournament.com/
[2]http://www.cs.mun.ca/ dchurchill/starcraftaicomp
[3]https://sites.google.com/site/starcraftaic/

that should reach the goals suggested by the higher level. These two levels are included into a goal-reasoning, planning and execution loop, which helps to control and process the real situation of the game to quickly update the plans.

This paper is organized as follows: first in Section II, we describe the planning framework. Then, Section III presents a description of the application domain. Section IV describes our contribution: the goal-reasoning process to control a StarCraft player. Later on Section V, we show preliminary experiments of our approach in different maps. Section VI contains the related work. Finally, in Section VII we present some conclusions and future work directions.

II. Planning Framework

In this section, we describe the different planning formalisms that define the Multilevel Planning approach. It considers two different kind of planning tasks: (i) a sequential classical planning task, which is encoded in the propositional fragment of the standard Planning Domain Description Language (PDDL) 2.2 [6]; and (ii) a temporal planning task, which is encoded in the propositional fragment of the standard in PDDL 2.1 [7]. In PDDL, a planning task is described in terms of objects of the world (units, buildings, mineral, etc), predicates which describe static or dynamic features of these objects (e.g. building are locations in a specific position), actions (a unit can move from one location to another, an unit can be training in a building), an initial state that describes the initial situation, and a goal definition which describes the objectives that must be reached (goals).

Definition 1: **(Planning task)**. A planning task can be defined as a tuple $\Pi = (F, A, I, G)$, where:

- F is a finite set of grounded literals (also known as facts or atoms).
- A is a finite set of grounded actions derived from the action schemes of the domain.
- $I \subseteq F$ is a finite set of grounded predicates that are true in the initial state.
- $G \subseteq F$ is a finite set of goals.

Any state s is a subset of facts that are true at a given time step. Each action $a_i \in A$ can be defined as a tuple $a_i = (Pre, Add, Del)$, where $Pre(a_i) \subseteq F$ are the preconditions of the action, $Add(a_i) \subseteq F$ are its add effects, and $Del(a_i) \subseteq F$ are the delete effects. $\mathit{Eff}(a) = Add(a) \cup Del(a)$ are the effects of the action. Actions can also have a cost, $c(a)$ (the default cost is one). An action a is applicable in s_i, if $Pre(a) \subseteq s_i$. The result of applying a in a state s_i generates a new state that can be defined as: $s_{i+1} = (s_i \cup Add(a) \setminus Del(a))$. A plan π is an ordered set of actions (commonly, a sequence) $\pi = (a_1, \ldots, a_n), \forall a_i \in A$, that transforms the initial state I into a state s_n where $G \subseteq s_n$. This plan π is valid if the preconditions of each action are satisfied in the state where the corresponding action is applied; i.e. $\forall a_i \in \pi, Pre(a_i) \subseteq s_{i-1}$ such that from applying the action a_i in the state s_{i-1} the system transits to s_i. The state s_0 represents I.

Definition 2: **(Temporal Planning task)**. A temporal planning task is a tuple $\Pi = (F, A^d, I, G)$ where

- F is a finite set of grounded literals (also known as facts or atoms) and numerical functions.
- A^d is a finite set of grounded durative actions.
- $I \subseteq F$ is a finite set of grounded predicates that are true in the initial state and a set of numerical functions.
- $G \subseteq F$ is a finite set of goals.

Each action $a_i \in A^d$ can be defined as a tuple $a_i = (D, Pre, Add, Del)$, where $D(a_i)$ is the duration of the action a. $Pre(a_i) = \langle Pre_s(a_i), Pre_o(a_i), Pre_e(a_i)\rangle$ where $Pre_s(a_i)$ are preconditions given at the start of a; $Pre_o(a_i)$ are preconditions given during the action's duration; and $Pre_e(a_i)$ represents preconditions given at the end of a. $Add(a_i) = \langle Add_s(a_i), Add_e(a_i)\rangle$, where $Add_s(a_i)$ are the add effects at the start of a_i and $Add_e(a_i)$, the ones at the end. $Del(a_i) = \langle Del_s(a_i), Del_e(a_i)\rangle$, where $Del_s(a_i)$ are the delete effects at the start of a_i and $Del_e(a_i)$, the ones at the end.

III. Description of the real time environment

Our approach has been applied to StarCraft: Brood War [4], which is a popular RTSVG that runs a science-fiction universe where three races (*Terran, Protoss* and *Zerg*) are at war. The aim is to defeat all your opponents. Players start the match in a random position of the map. During the match, the player exploits the resources in order to (i) train new workers; (ii) build new buildings - which allow players to train soldier units, unlock stronger units and unlock new buildings-; and (iii) research new technologies - improve units' features or unlock new skills-. Besides, the player has to explore the environment to discover new resources, expand its territory and find the opponents' position. In the end, players must define a global strategy in order to destroy every opponent's units and buildings.

Our definition of global strategy is composed by simple actions. First, actions related with units' control (e.g. collect, move, train, build, explore, research and attack) for micromanagement. Second, actions related with the global strategy (e.g. which kind of unit must be trained, which kind of building must be built, when and where to attack) for macromanagement. For instance, in order to build a tank we need first to build a *Machine shop*, a *Factory* and a *Command Center*. Thus, we need at least one worker to build the four buildings on four suitable locations in the map -that need to be explored first-. For instance, when the action "*train a tank*" is executed, a set of micromanagement actions were performed first. From the planning perspective, the macromanagement actions can be considered as dynamic soft-goals, which are generated during the match according to the information provided by the environment and the player; and the micromanagement actions are the sequence of actions of the plan π that reach the soft-goals. that

IV. Goal-reasoning through Multilevel Planning

StarCraft is a RTSVG where two or more players must compete for the resources of the environment in order to win

[4] https://starcraft.com

the match. From the point of view of planning, a StarCraft player should execute a sequence of actions – plan – provided by the planner to defeat all its opponents. However, generating a sequence of actions in a dynamic environment with partial information arises some issues: (i) new information about the environment is usually discovered during the action's execution; (ii) actions must be quickly chosen to interact within the environment as fast as possible - the sooner it happens, the greater the similarity with the current planning state; and (iii) goals should be generated dynamically during the game because of the lack of information. As we described before, the main goal of StarCraft is to win the match. Thus, the first issue we faced is how to model the PDDL domain to reach this hard goal. We are certain that this could not be satisfied if we just used classical planners. As the goal *win the match* is an abstract goal, it cannot be reached from a specific action executed by player. Neither a group of actions executed sequentially or in parallel guarantees to win the math. As the environment changes fast, it makes useless any long-term plan.

Our approach solves this issue by having a process that generates soft goals. They could be easily reached (or might not) according to the state of the match. Also, the agent has "high-level" tasks such as move, build, train or attack. Those can work as soft-goals as long as there is some low-level process to handle them on the game.

In previous works, the goal-reasoning problem has been addressed from different directions: (i) using an independent goal-reasoning system which receives a model of the environment, the current state and at least an initial goal, and returns either the same goal or a new one [14]; (ii) learning a model by means of Machine Learning to predict which goals must be achieved in order to control a real-time system [18]; and (iii) performing a hard goal life-cycle composed of different steps (selection, expansion, commit, dispatch, and finally execution) [10], [21]. In this paper, we propose an autonomous approach based on multiple levels of planning [17] that encapsulates the complexity of the domain in different layers.

The multilevel planning system is composed by two levels (macro and micro) of planning. The macromanagement level (high level) employs an abstract representation of the environment (numerical and logical) in order to define the set of soft-goals. This level uses temporal planning, which provides a richer version of PDDL e.g durative-actions, numerical fluents in preconditions and preconditions, etc. The output plan is a parallel plan, which explicitly indicates when two actions can be executed in parallel. Additionally, the soft-goals generation allow to use different policies for the generation of the plan in the micromanagement level. Figure 1 shows an example of an action defined in temporal PDDL which generates a goal `build-building(?b, ?l)` where `b` is the type of the building and `l` is the technological level needed to build it. As a result, this action generates goals to build any kind of building except for *Refinery* and *Supply Depot* which have special properties.

The temporal planner needs three elements to work: (i) an initial state; (ii) a specific domain that accurately describes a set of actions that, when instantiated, will serve as goals to "guide" the low-level planner; and (iii) a set of goals in order to stop the search process when they are satisfied. We were inspired by the limited horizon search approaches [11], [13] to define the stop criteria, which in our case is the upper-bound of goals that can be generated. We defined a fluent called *(max-number-goals)* as a counter that decreases in one each time an action-goal is generated. It stops in zero. The sequence of actions generated by the temporal planner is transformed into goals, which are used by the low planning level. Each action of the plan is considered like a predicate which must be reached in the next level.

```
(:durative-action generate-build-building
 :parameters (?b - building
              ?l - level
              ?a - area)
 :duration (= ?duration
              (* (building_cost ?b)
              (number_buildings ?b ?a)))
 :condition (and
    (at start (> (free_units scv) 0))
    (over all (> (max_number_goals) 0))
    (over all (tech_level ?l))
    (over all (tech_level_building ?b ?l))
    (over all (can_build ?b)))
 :effect (and
    (at end (decrease (max_number_goals) 1))
    (at start (decrease (free_units scv) 1))
    (at end (increase (free_units scv) 1))
                      (at end (have_building ?b ?a))
    (at end (increase (number_buildings ?b ?a) 1))
    (at end (increase (goals_score)
                      (score_building ?b)))))
```

Fig. 1. Example of a durative action that generates building goals.

The micromanagement level (low level) uses a structured representation of the environment to reach the previous soft-goals. This level employs classical planning in order to reach the goals. The plan described in Figure 3 is transformed into a group of three goals: (1) train a worker *(SCV)* in a *Command Center*, (2) build *Barracks* and (3) build a *Supply Depot*. Each goal needs a sequence of detail actions to be reached. The training goal needs two actions: one to chose a building of type command center (if there are more than one) and start training process of a *SCV* unit. After that, each building goal needs four actions to be reached: one action to choose an available worker that will perform the building process, a second action that chooses an available location using a external search algorithm, a third action that moves the worker to the building location and finally a fourth action which starts the building process. The agent will interleave goal and plan generation in a infinite loop using the PELEA architecture [5], which now implements the macro and micromanagement levels into its different modules. Therefore, in order to handle more complex situations we are using different planners working at

[5]The architecture described here is an instantiation of the original version [19] which employs two different representations of the environment in PDDL.

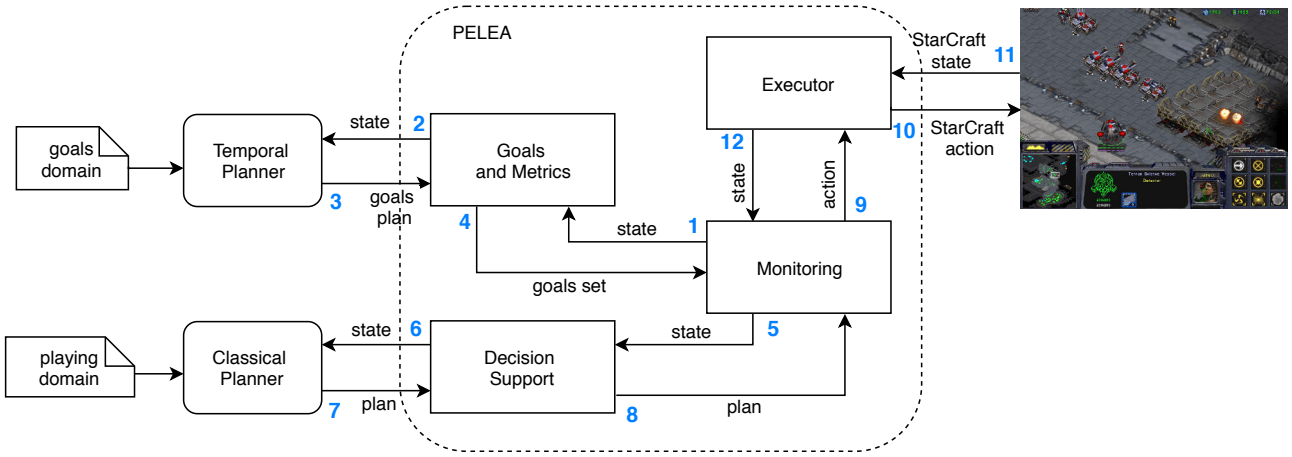


Fig. 2. Overview of the PELEA architecture applied to StarCraft. The multilevel planning approach is shown on the left of the Figure, where the temporal planner is in charge of generating the goal-based actions' plan (macromanagement level) and the classical planner generates the sequence of game actions (micromanagement level). Numbers in blue indicate the flow of the architecture from the initial state of the game. First, *Monitoring* asks to the *Goals and Metrics* for the high-level plan (initially the set of goals is empty); then the plan is transformed into a set of goals and is sent to the *Decision Support*, which generates a low-level plan based on those goals. The resulting plan is sent back to the *Monitoring*. This module sends the plan to the *Executor* action by action. When each action is executed, the game-state is updated and processed.

```
0.000: (train-unit scv command_center tl_one)  [7.0]
7.001: (build-building barracks tl_one)  [22.0]
29.002: (build-building supply_depot tl_one)  [8.0]
```

Fig. 3. Example of a high-level plan, whose actions will be used as the goal set for the low level.

different levels of abstraction. This version integrates planning, execution, monitoring and goal generation. Figure 2 shows the structure of the agent which is composed of four sub-modules.

- **Monitoring**: it is the central module of the planning agent. It synchronizes the communications among the other modules and monitors the action's execution i.e. dispatches the next action to *Execution* Module, requests for a new plan to the *Decision Support* and checks for differences between the expected state and the observed state of the environment sent by *Monitoring*. If an observed state is not valid, this module has to start another planning episode to generate a new plan according to the observed state.
- **Executor**: this module runs the actions into the environment by means of the *Brood War Application Programming Interface* (BWAPI)[6]. BWAPI provides a Java Native Interface to the Brood War API that uses the shared memory bridge to give our code access to the game state. Information that can be easily extracted through the API such as units' state, resources location; or some orders can be issued through training or moving commands. On every frame, JNI-BWAPI sends a game state update to the Java AI agent and waits for a response, which will contain a set of commands to execute. Besides, this module is responsible for initiating the goal-reasoning, planing and execution loop by sending to the *Monitoring* a particular problem and domain definition to be solved. Both the problem and the domain are described in PDDL.
- **Decision Support**: this module generates a plan of actions by the invocation of a classical planner (Metric FF [9]). When the *Monitoring* informs about a discrepancy between the observed state and the expected planning state, the *Decision Support* invokes the classical planner to generate a new plan.
- **Goals & Metrics**: this module searches and suggests different reachable goals according to the available information about the environment by using a temporal planner (OPTIC [2]). Additionally, this module keeps a registry of the different goals that have been solved in order to generate new goals when all of them have been reached.

A. Planning, Monitoring and Execution Architecture

We have developed a Planning and Execution agent based on PELEA. We contribute with an extended version of the *Goals and metrics* module

[6]https://code.google.com/p/jnibwapi/

B. Modeling StarCraft with PDDL

StarCraft uses a lot of complex (numerical and logical) information about the players and the environment to describe the current state of the game. We have described the environment using two abstraction levels that are modelled in PDDL. Our representation of the game state is modeled by using a sequence of binary and numerical values which represent the number of units, buildings and resources available for the players (our player and the opponents) located in the different areas of the map. The specific location of the units or their life points is not relevant for the planning process. Thus, they are omitted.

The game state is described using a sequence of variables (numerical and binary) which represent the units of each player. The map environment is divided into areas of same size. Commonly each area has a set of resources which can

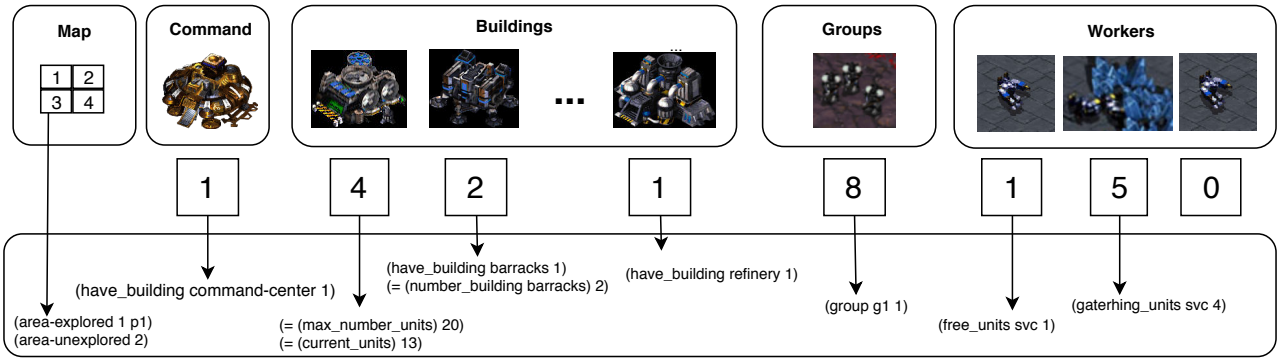


Fig. 4. Transformation of the partial state of the world from StarCraft to PDDL. The numerical sequence of values represents the current state of the game. Positions on the sequence correspond to one of the categories on the top of the Figure. Numbers are mapped into PDDL predicates as in the bottom part of the Figure.

be exploited. The units and buildings are associated to an area in order to have an approximate location of the different units. Figure 4 shows an example of a partial representation of the numeric state, which is later transformed into a PDDL state. In this case, the units of the player are only located in one area and there is not information available about the enemies yet. Each numerical variable is transformed in a set of logical predicates or numerical functions which are included into the game state depending of the abstraction level. The macromanagement level uses some logical predicates, the numerical functions and a group of special functions which can be tunned in order to change the goal generation process: (i) `(goals-score)` which computes the game score of the plan; (ii) `(total-time)` which computes an approximation of the total time to reach the goals; and (iii) `(max-number-goals)` which defines the maximum number of goals that can be generated. The micromanagement level uses only the logical predicates in order to simplify the plan generation. Figure 4 shows an example of the partial representation of the game state, where there is a *Barracks* building which is built in the first area of the map. Thus, the location of each building is going to be represented using two predicates: (i) `(have_building_barracks 1` which means that there is at least one barracks in the area 1 of the map; and (ii) `(number_building barracks 1)` which means which number of barracks are in the area 1. Besides, there are some other predicates that indicates if the agent can be built `(can_build ?building)` or if the technological level of the agent allows to build that specific building `(tech_level_building ?building ?level)`.

As a result, we have come up with 10 directed actions and 4 derived actions for the micromanagement level. These actions are used by the classical planner to later perform most of the available game behaviors using the goals generated by the macromanagement level. The directed actions (*move a unit, move a units' group, gather, build, train, research, attack, quarter and explore*) produce changes into the environment and the derived actions (*find-unit, find-building-location, find-explore-location and find-attack-location*) collect information to later execute a directed action.

V. Experimental Results

This section presents preliminary results when using the Planning and Execution architecture described in Section IV to play on different scenarios from StarCraft. The Brood War Application Programming Interface (BWAPI) was used to connect the executor to the StarCraft game in order to send and receive real-time information from the game. The experiments were conducted on an Intel Core i7-6700T 2.80 GHZ (32 bits) with 3.50 GB running on Windows 7 for the StartCraft simulator and on and Intel Core i7-6700T 2.80 GHZ (64 bits) with 2 GB running on Ubuntu Linux. The maximum planning time for the low-level planner to solve a problem has been set up to 20 seconds and the horizon `(max-number-goals)` for the high-level planner has been set up to 5 and 3 goals. The high-level planner is OPTIC [2] while the low-level planner is METRIC FF [9]. Each scenario has been played 5 times until the game is finished. Initially we tried to compare our approach with some other approaches, but either they had been developed for other platforms or they needed some kind of previous human-directed learning to work properly.

Table I shows the results of playing on StarCraft with our Multilevel Planning approach. We chose three different scenarios (Astral Balance, Baby Steps, Ice Mountain) where three different strategies (greedy, global, parallel) have been deployed. We have used the Terran race. In general, the Multilevel Planning approach can suggest different set of goals and use them to generate short plans to execute actions in the environment. There are two important aspects to analyze in the results: (i) the planning time; and (ii) the number of planning steps. Besides, we have defined three goal-selection policies in order to analyze the influence of the different goals in the planning and execution process: (a) Greedy goal selection, which generate a plan for each goal; (b) Global goal selection, which generate a plan for all goals; and (c) Parallel goal selection, which extracts a subset of goals that can be reached in parallel. Two goals are considered parallel if the

TABLE I
COMPARING THREE GOAL-SELECTION STRATEGIES OVER THREE DIFFERENT MAPS FROM STARCRAFT. THE PERFORMANCE OF THE DIFFERENT STRATEGIES HAVE BEEN MEASURED OVER SIX METRICS. GOALS CORRESPONDS TO THE NUMBER OF GOALS GENERATED DURING THE MATCH. PLANNING STEPS CORRESPONDS TO THE NUMBER OF PLANNING EPISODES. UNITS CORRESPONDS TO THE NUMBER OF UNITS TRAINED. BUILDINGS CORRESPONDS TO THE NUMBER OF BUILDINGS BUILT. PLANNING TIME CORRESPONDS TO THE AVERAGE TIME OF THE PLANNING EPISODES. TIME CORRESPONDS TO THE TOTAL MATCH TIME.

Map	Strategy	Goals	Planning steps	Units	Buildings	Planning time	Time
Astral Balance	Greedy	18	17	8	4	0.3	5:45
Astral Balance	Global	27	6	18	7	3.55	4:36
Astral Balance	Parallel	25	7	17	6	2.94	5:25
Baby Steps	Greedy	16	15	7	4	0.1	4:38
Baby Steps	Global	31	8	20	6	3.12	5:40
Baby Steps	Parallel	29	7	16	7	1.12	5:15
Ice Mountain	Greedy	15	13	9	3	0.4	3:18
Ice Mountain	Global	28	6	15	8	3.55	4:07
Ice Mountain	Parallel	26	9	17	7	2.34	3:56

game actions from which they are derived can be executed in parallel.

On the one hand, the use of methods that generate a larger goal set (global selection and parallel selection) decreases the number of planning steps but increases the planning time on each iteration. The complexity of the problem to solve is increased by the number of goals. On the other hand, the use of a greedy method that generates a small goal set decreases the planning time but increases the number of planning steps making the agent work slower and decreasing the number of actions per iteration that are executed in the environment.

VI. Related work

The approach described in this paper is focused on applying Automated Planing (AP) as a reasoning model for a StarCraft autonomous player. There are previous approaches that have used either some other AI techniques for reasoning or AP. SOAR [12] is a cognitive architecture that implements state abstractions, planning and multi-tasking based on Finite State Machines (FMS). This architecture was developed using two layers. First layer is a middleware layer that serves as the perception system and gaming interface. Second layer is composed by a set of FSMs that executes parallel actions into the game according to the information obtained from the first layer.

Darmok [16] is an architecture that implements on-line case-based planning by learning from demonstration on WARGUS (an open source clone of WarCraft II). This architecture employs a planning, execution and monitoring loop, similar to the one used in our approach. The execution cycle analyzes if the last action is finished, updates the current state and picks up the next action. However, Darmok must be first trained by playing a collection of games to generate the case library. In [22], authors present a bot that uses a reactive planning implementation of the Goal-Driven Autonomy (GDA) [15]. UAlbertBot [4] uses a heuristic search algorithm to find concurrent plans of actions which are constrained by unit dependencies and resource availability. Theses plans are focused on creating a certain number of units and structures in the shortest possible time span.

The majority of these approaches use control systems for videogames agents, but only one of them plays on Startcraft. Our approach mainly differs from those previous works in the generation and achievement of goals, which is the main focus of this paper. We do not use any training phase to learn.

VII. Discussion and Future Work

In this paper, we propose a Multilevel Planning approach to perform goal-reasoning and planning on RTSVGs using two different levels of planning. This approach has been deployed using the general-purpose software architecture (PELEA) in order to implement an autonomous player for the video-game StarCraft: Brood War. The preliminary results presented in this paper allow us to draw some very interesting conclusions: (i) it is possible to use AP for goal reasoning according to the information about the environment; (ii) divide the complexity of the environment on different levels decreases the complexity of the planning task and the definition of the domain; and (iii) using a temporal planner to generate a sequence of actions, which is going to be transformed into goals, allows to decrease the information used for planning in the second level. Our contribution presents some advantages: it can be easily improved by adding new actions/goals/information on the domain and being easily adapted to different RSTVGs, as the only domain-dependent parts are the Executor module and the planning domains.

While our initial results are encouraging, there is room to improve the approach presented in this paper. On one hand, the different domains used for planning and goal-reasoning can be refined in order to include more actions to generate new goals or more complex plans of actions. Besides, it is possible to define different domains for the other races of the video-game. On the other hand, we can explore other techniques based in Machine Learning to replace both the goal-reasoning or the planning process: (i) learning goal trees to suggest goals; and (ii) using case-based planning to learn previous plan and avoid some planning and replanning episodes. We could also study HTN planning to analyze its flexibility when working on a dynamic environment; it could complement or replace one of the planners.

Acknowledgments

This paper has been partially supported by the European Union's Horizon 2020 Research and Innovation program under Grant Agreement No. 730086 (ERGO) and the Spanish MICINN project TIN2017-88476-C2-2-R.

References

[1] Mitchell Ai-Chang, John L. Bresina, Leonard Charest, Adam Chase, Jennifer Cheng jung Hsu, Ari K. Jónsson, Bob Kanefsky, Paul H. Morris, Kanna Rajan, Jeffrey Yglesias, Brian G. Chafin, William C. Dias, and Pierre F. Maldague. Mapgen: Mixed-initiative planning and scheduling for the mars exploration rover mission. *IEEE Intelligent Systems*, 19(1):8–12, 2004.

[2] J. Benton, Amanda Coles, and Andrew Coles. Temporal planning with preferences and time-dependent continuous costs. In *ICAPS*, 2012.

[3] Sara Bernardini, Maria Fox, and Derek Long. Planning the behaviour of low-cost quadcopters for surveillance missions. In *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS)*, Portsmouth, New Hampshire, USA, 2014.

[4] David Churchill and Michael Buro. Build order optimization in starcraft. In *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2011)*, pages 10–14, 2011.

[5] Marc de la Asunción, Luis A. Castillo, Juan Fernández-Olivares, Óscar García-Pérez, Antonio González, and Francisco Palao. SIADEX: an interactive knowledge-based planner for decision support in forest fire fighting. *Artificial Intelligence Communications*, 18(4):257–268, 2005.

[6] Stefan Edelkamp and Jörg Hoffmann. Pddl 2.2 : The language for the classical part of ipc-4. In *Proceedings of the fourth International Planning Competition. International Conference on Automated Planning and Scheduling*, Whistler, British Columbia, Canada, 2004.

[7] Maria Fox and Derek Long. Pddl2.1: An extension to pddl for expressing temporal planning domains. *JAIR*, 20, 2003.

[8] Javier Garca, Jos E. Flrez, lvaro Torralba Arias de Reyna, Daniel Borrajo, Carlos Linares Lpez, Angel Garca Olaya, and Juan Senz. Combining linear programming and automated planning to solve intermodal transportation problems. *European Journal of Operational Research*, 227(1):216–226, 2013.

[9] Jörg Hoffmann and Bernhard Nebel. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

[10] Benjamin Johnson and Mark Roberts. Goal reasoning with informative expectations. In *Proceedings of the Fourth Annual Conference on Advances in Cognitive Systems (ACS-16)*, 2016.

[11] Richard E. Korf. Real-time heuristic search. *Artificial Intelligence Journal*, 42(2-3):189–211, 1990.

[12] John E. Laird, Allen Newell, and Paul S. Rosenbloom. Soar: An architecture for general intelligence. *Artificial Intelligence*, 33(1):1–64, sep 1987.

[13] Moisés Martínez, Fernando Fernández, and Daniel Borrajo. Planning and execution through variable resolution planning. *Journal of Robotics and Autonomous Systems*, In press.

[14] Matthew Molineaux, Matthew Klenk, and David W. Aha. Goal-driven autonomy in a navy strategy simulation. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, 2010.

[15] Héctor Muñoz Avila, Ulit Jaidee, David W. Aha, and Elizabeth Carter. Goal-driven autonomy with case-based reasoning. In *Proceedings of the 18th International Conference on Case-Based Reasoning Research and Development*, ICCBR'10, pages 228–241, Berlin, Heidelberg, 2010. Springer-Verlag.

[16] Santiago Ontañón, Kinshuk Mishra, Neha Sugandh, and Ashwin Ram. Case-based planning and execution for real-time strategy games. In *Proceedings of the 7th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR '07, pages 164–178, Berlin, Heidelberg, 2007. Springer-Verlag.

[17] Alison Paredes and Wheeler Ruml. Goal reasoning as multilevel planning. In *Proceedings of the ICAPS-17 Workshop on Integrated Execution of Planning and Acting (IntEx-17)*, 2017.

[18] Alberto Pozanco, Susana Fernandez, and Daniel Borrajo. Urban traffic control assisted by ai planning and relational learning. In *Proceedings of the 9th International Workshop on Agents in Traffic and Transportation (IJCAI'16)*, 2016.

[19] Ezequiel Quintero, Vidal Alcázar, Daniel Borrajo, Juan Fernández-Olivares, Fernando Fernández, Angel García Olaya, César Guzmán, Eva Onaindia, and David Prior. Autonomous mobile robot control and learning with the pelea architecture. In *Proceedings of the AAAI-11 Workshop on Automated Action Planning for Autonomous Mobile Robots (PAMR)*, San Francisco, CA, USA, 2011.

[20] K. Rajan, C. McGann, F. Py, and H. Thomas. Robust mission planning using deliberative autonomy for autonomous underwater vehicles. In *Proceedings of the Workshop on Robotics in Challenging and Hazardous Environments, ICRA*, Rome, Italy, 2007.

[21] Mark Roberts, Vikas Shivashanka, Ron Alford, Michael Leece, Shubham Gupta, and David W. Aha. Goal reasoning, planning, and acting with actorsim, the actor simulator. In *Proceedings of the Fourth Annual Conference on Advances in Cognitive Systems (ACS-16)*, 2016.

[22] Ben Weber, Michael Mateas, and Arnav Jhala. Building human-level ai for real-time strategy game. In *Proceedings of AIIDE Fall Symposium on Advances in Cognitive Systems*, 2011.

A Graphic User Interface for Images Edge Detection. A Proposal to Combine Ant Colony Systems and Fuzzy Logic

M. Angélica Pinninghoff J.
University of Concepción
Concepción, Chile
mpinning@udec.cl

Sebastián Espinoza J.
University of Concepción
Concepción, Chile
siespinozaj@gmail.com

Ricardo Contreras A.
University of Concepción
Concepción, Chile
rcontrer@udec.cl

***Abstract*—The problem of band detection in DGGE images is a key issue in biology because it allows for a correct identification of microbial populations present in biological samples. These particular images present characteristics like image distortion or the presence of noise, that affect the identification process. Traditional mechanisms for edge detection and commercial softwares only partially solve this problem. This proposal introduces a graphical interface which combines ant colony optimization and fuzzy logic to generate a map of edges for classifying pixels that are part of an edge and pixels that are not. The hybrid system is supported with an alternative matrix of pheromone and introduces an alternative set of fuzzy rules.**

***Index Terms*—graphic interface, edge detection, ACO systems, fuzzy logic**

I. Introduction

In artificial vision and in image processing, edge detection deals with the localization and identification of significant gray level variations in a digital image. Localization refers to the search of points at a particular location in a grid of pixels. Identification refers to the process of deciding whether a particular pixel belongs to an edge.

In image processing, an important number of edge detectors have been proposed, exhibiting differences in terms of mathematic and algorithmic properties [9]. One of the standard edge detection methods is proposed by Canny [7], that offers a very effective pixel identification and analyzes every pixel in the image.

Edge detection is carried out through an evaluation of the differences of pixels brightness intensities. The edge detection process is performed by evaluating differences in brightness intensity among pixels that belong to different regions in the image. A region can be characterized as edge or non-edge. It is usual to pre-process an image in such a way that the image contains only the brightness intensity information. These images are known as gray-scale images, in which every pixel holds a value between [0, 255], representing different gray intensity values, from black to white. Usual preprocessing mechanisms are edge thinning or erosion [3]. The effect of erosion on grayscale images is to erode away the boundaries of regions of foreground pixels (i.e. white pixels typically). Thus areas of foreground pixels shrink in size, and holes within those areas become larger.

Depending on the chosen method and depending on the characteristics of the image, it is possible to identify a pixel as part of an edge when the pixel is actually part of the background of the image. This problem is known as *false positive detection*. Typical characteristics of an image that lead to false positive detection are noise, blurring or lack of contrast between neighbor regions.

When dealing with brightness intensities of pixels as numerical values, it is possible to apply mathematical operations on pixels, to decide whether a particular pixel belongs to an edge. Canny algorithm, one of the best known algorithms for detecting edges, computes the gradient for a pixel (the difference of intensities among a pixels and their neighbors). The requirement of applying a gaussian filter to preprocess the image, has triggered the search for alternative methods to extend the Canny algorithm.

One of the research directions to complement the Canny algorithm, is the procedure known as *Ant Colony Optimization* (ACO), based on the ants behavior in nature [10]. ACO refers to the cooperative working of ants when foraging for food. Although communication between ants during such search is limited, the colony operates far more successfully as a whole than any individual ant within it [16]. Artificial ants manage to establish shortest routes between the colony and a food source. This capability depends upon the production of pheromones and laying of a pheromone trail by individuals ants during their search for a food source. ACO is a population-based metaheuristic that mimics the foraging behavior of ants to find approximate solutions to difficult optimization problems.

ACO has been applied to the edge detection problem, as in the work of [19]. Authors establish a pheromone matrix that represents the edge information presented at each pixel position of the image, according to the movement of a number of ants which are dispatched to move on the image. The movements of these ants are driven by the local variation of the image's intensity values.

The work of [18] proposes an approach that introduces fuzzy logic to decide, with simple rules, the ant's movements

based on heuristic information, with a dynamically updated influence of the pheromone and heuristic information. Authors claim that results are improved with respect to previous experiences that use ACO. This work is important because it gives birth to a new category of edge detectors, that combines optimization through ACO and fuzzy logic inference [20]. This new category is a hybrid ACO-Fuzzy logic system.

In recent years, different researchers have explored the use of fuzzy logic in image processing. In [1] authors propose the use of fuzzy logic for the automatic analysis of X-ray images of industrial products for defect detection. The architecture is based on fuzzy logic and authors claim it is quite tolerant to imprecise input data, and therefore insensitive to noise. In [2], different implementations of the fuzzy system as an edge detector had been overviewed and compared. According to the author, the methods presented shows a better capability to recast the image to new grey level and provide a better tool to get better results in the field of the image edge detection.

In [15], Nawgaje et al. present a Fuzzy Inference System approach to detect the edges of the microscopic images within colour, which is robust and has stability degrees. They proposed the logic based technique which is a set of three pixels and, using the smallest mask of 2*2 window image, consists of a set of fuzzy rules which highlight all the edges that are correlated with an image.

In [22], authors present the implementation of a simple, flexible and efficient fuzzy logic based algorithm to detect the edges of a vehicle in an input image by scanning it through the 2*2 mask. The 2*2 masks is slid over entire vehicle image, and then pixel values of masks are examined through various rules. Based on these set of rules it is decided that particular pixel is edge or not. Peric, in [17] presents a fuzzy set based approach on edge detection, applied to region labeling, a process by which the digital image is divided into units and each unit is given a label (sky, grass, house, and so on).

In [12], authors propose a fuzzy logic based edge detection algorithm for noisy images. The proposed algorithm is based on a 3x3 window mask and fuzzy rules. The window mask and fuzzy rules are defined in a manner such as to detect edges in both noise free and noisy images. The algorithm was tested on grayscale images of size 512x512 pixels, and authors claim it detected all the edge pixels in noisy free and noisy grayscale images. Authors are currently working to extend the method to clinical examination of noisy Magnetic Resonance Images (MRI).

In [13], authors combine fuzzy logic with random walker method (a supervised segmentation method) to make resulting segmentation better in texture and quality. Fuzzy rules are used to approximate boundaries in images which improve segmentation results, when dealing with medical images. The objective of this work is to use a hybrid ACO-Fuzzy logic approach to solve the problem of edge detection on DGGE (Denaturing Gradient Gel Electrophoresis) images.

This article is structured as follows, the first section is the present introduction, the second section describes the theoretical frame supporting this work, the third section describes design and implementation issues. Section four presents results obtained and the final section shows the conclusions.

II. Theoretical frame

In this section, there is a summarized introduction to fundamental concepts involved in this work. While digital images provide the core subject, the main focus is in DGGE images. Then, there is a short introduction to fuzzy logic concepts, and the metaheuristic based on ants behavior: ACO.

A. Digital images

A digital image is a discrete representation of a two-dimensional image through the use of an image function $f(x, y)$, where x and y represent coordinates in a two-dimensional space. This function indicates the intensity or level of gray that a particular point in the image (known as pixel) presents. As x and y represent discrete locations, a matrix is the usual mechanism for representing a grayscale image.

Information that can be obtained from the matrix include the image resolution (size of the matrix, or number of pixels that define the image) and the color depth, which is the number of color values that can be assigned to a single pixel in an image. As an example, in binary images every pixel is represented by only one bit, which indicates white color, if the value is 1 and the black color if the value is 0; if we use 8 bits for each pixel, we can obtain 256 values in a gray scale ranging from 0 (representing black) to 255 (representing white). Edge pixels are pixels at which the intensity of an image function changes abruptly, and edges (or edge segments) are sets of connected edge pixels. On the other hand, edge detectors are local image processing methods designed to detect edge pixels.

B. DGGE images

Denaturing Gradient Gel Electrophoresis (DGGE) [5], [14], is a DNA-based technique which generates a genetic profile or *fingerprint* which can be used to identify the dominant members of a microbial community. DGGE has been used to investigate microbial responses in a wide variety of applications, including bioremediation assessment, wastewater treatment, drinking water treatment, biofilm formation, microbial induced corrosion, among others.

DGGE separates mixtures of amplified 16SrRNA gene segments, which are all the same size, based on nucleotide sequence. Denaturing breaks apart the two strands of the DNA molecule. Gradient Gel is a gel with an increasing concentration of a chemical (denaturant) which breaks apart the DNA molecule. Electrophoresis is the application of an electric current across a gel. In response to the current, double-stranded DNA migrates (moves down) the gel. Denaturing the DNA molecule forms Y and T-shaped structures greatly slowing migration. Finally, this process allows to obtain an image composed of bands and lanes [4].

Lanes are the vertical columns shown in Figure 1 and each one of them represents a DNA sample, except the reference lanes which are the leftmost and the rightmost lanes. Reference

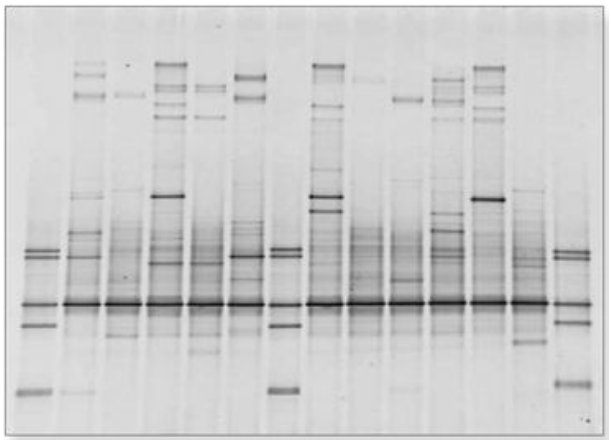


Fig. 1. A typical DGGE image.

lanes are used to indicate the molecular weight, measured in base pairs (*bp*) of the DNA.

C. Fuzzy logic

In the 60s, Zadeh [23] introduced the concept of partial set membership, to provide a reasoning mechanism that could use fuzzy variables, i.e., variables that define the language subsequently used to discuss a fuzzy concept such as temperature, pressure, age.

According to fuzzy set theory, a fuzzy set **A** on a universe of discourse **U** is characterized by a membership function $\mu_A(x)$ that takes values in the interval [0, 1]. Fuzzy sets represent commonsense linguistic values. A given element can be a member of more than one fuzzy set at a time. A fuzzy set **A** in **U** may be represented as a set of ordered pairs, with each pair consisting of a generic element x and its grade of membership function:

$$\mathbf{A} = \{(x, \mu(x)) \mid x \in \mathbf{U}\} \quad (1)$$

Knowledge is represented in the form of If ... Then rules, and these rules do not work with precisely defined values. Each variable is assigned a fuzzy value such as *high*, *moderate*, *advanced*, etc. These fuzzy values cover a range of measured values.

Our everyday language is full of fuzzy descriptors called linguistic variables. For example, height is often measured in centimeters and tall and short describe regions within this continuous scale. A linguistic variable such as age may have a value such as young, or its antonym, old. However, the great utility of linguistic variables is that they can be modified via linguistic hedges applied to primary terms. These linguistic hedges can be associated with certain functions. Being fuzzy means that there is no clear boundary between the end of one value and the start of another.

Linguistic values are context dependent, in that the range of values they are defined over depends on the variable with which they are associated. Contextual issues for any application are taken into account with a function that is defined for each linguistic variable value. The purpose of a function is to convert a measured value into a linguistic value (for example, 0 Celsius degree into *freezing*).

Fuzzy rules represent control knowledge and the task of inferencing is to map a series of input variables to a controlling output variable. The mapping from a measured value to a linguistic value is done using a fuzzy membership function. A membership function exists for each linguistic variable value, and the output of that function is a degree of membership that measures the strength of association that a measured value has with a linguistic variable [6].

D. Ant Colony Optimization

Ant Colony Optimization (ACO) algorithms are metaheuristic techniques usually applied to optimization problems. These approaches are based on the behavior of some foraging ant species, which exhibit the capability of finding the shortest path between the nest and the source of food. Ants do not show an individual behavior, but a collaborative one, under an indirect communication system, by using a specific substance, called pheromone, that ants deposit on the trails they traverse. Ants detect the presence of pheromone and tend to follow trails in which pheromone concentration is higher, which indicates that those trails have been often traversed by other ants.

This ant behavior was studied by Deneubourg [8] and based on these models Dorigo et al. [10] implemented an artificial model to emulate the ants behavior as described by Deneubourg. Results let the researches conclude that the artificial behavior of artificial ants is similar to the natural ant behavior and Dorigo proposed a derivative model for solving optimization problems [11].

Many of the computational problems that belong to the NP-hard class can be seen as optimization problems, and hence it is possible to apply ACO algorithms to obtain solutions that are close to the optimum. Typical examples are the vehicle routing problem, task assignment, and scheduling. Ant-based algorithms are also applied to telecommunications problems and industrial applications [10].

III. Design and implementation

The language chosen for development is Python in a development environment PyQt (based on Qt). The language is a software of free distribution and offers a wide library for scientific applications. At the same time, the language supports the implementation of high user-interaction applications. The software is intended for use by researchers in the Artificial Intelligence and Bioscience domains.

A. Hybrid Model

The proposed hybrid system in this work is modelled after the one proposed by Tyagi et al. [18], which incorporates ACO and Fuzzy Inference techniques in its behavior. Figure 2 illustrates the general structure of the model. The input image is subjected to grayscale conversion and gaussian filtering as part of the pre-processing stage. Later, this image feeds the Fuzzy Inference component, which characterizes the quality

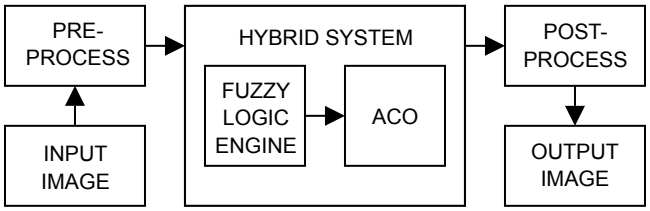


Fig. 2. General structure.

of row, column, or diagonal in its fuzzification step for each pixel in the image; this gives the name to the corresponding fuzzy variables: Mrow, Mcol and Idiag. These variables are further categorized into Low, Medium and High values, with each being modelled as a gaussian function parameterized by center (μ) and width (σ). Lastly, the fuzzy rules defined in this component determine the output image resulting from the application of a Mamdani-type defuzzification step on each pixel, where the Edgeness output variable follows the same categorization scheme as the three input variables described previously.

Once generated, the output image from the preceding stage provides the heuristic data, or Heuristic Matrix, for the ACO component in the model. In this component, a number of artificial ants (m) are randomly distributed throughout a uniformly weighted (τ_0) matrix of ones, called Pheromone Matrix, of the same dimensions as the input image. For a number of cycles (n), each ant, selected at random, moves a number of contiguous steps (s) according to the neighboring pixel (set **N**) that maximizes the value in Equation 2, where the pheromone (α) and heuristic (β) exponents vary after an ant's movement from pixel i to k depending on the difference (Δ) between the minimum and maximum values of its neighbors in the Heuristic Matrix:

$$i_{next} = argmax\left(\{\tau_k^{\alpha}\eta_k^{\beta} \mid k \in \mathbf{N}\}\right) \tag{2}$$

This movement produces a pheromone deposition in the ant's destination proportional to the corresponding value (η) in the Heuristic Matrix. The ants are randomly redistributed on the Pheromone Matrix after the end of each cycle. Finally, the process is subjected to two pheromone evaporation phases: a local one (ρ) after the step movement of each ant (Equation 3), and a global one (ψ) after the completion of each cycle (Equation 4). The output image of this stage is the resulting Pheromone Matrix.

$$\tau_i \leftarrow (1-\rho)\tau_i + \rho\eta_i \tag{3}$$

$$\tau_i \leftarrow (1-\psi)\tau_i + \psi\tau_0 \tag{4}$$

The last stage in the model is post-processing, where each pixel in the produced image is classified into border or non-border by thresholding. In line with other work using the Tyagi et al. Hybrid Model, such as [21], a modification is introduced in the form of an alternative initial Pheromone

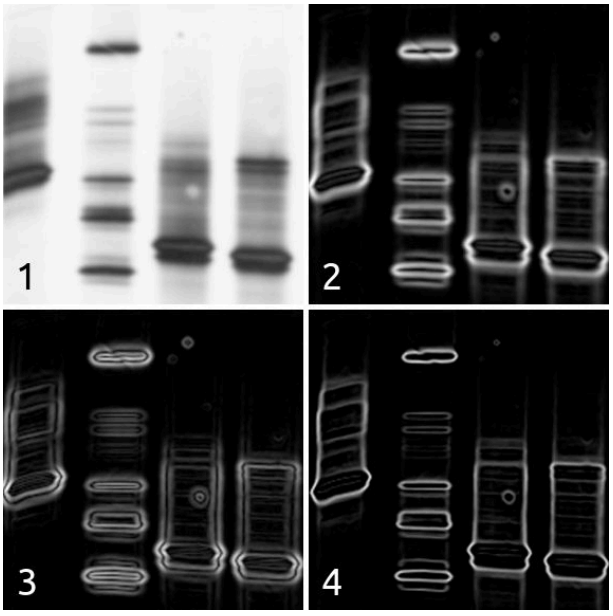


Fig. 3. Effect of substracting modified laplacian from gradient. 1: original, 2: gradient, 3: modified laplacian, 4: substraction. Note the formation of creases in the modified laplacian.

Matrix in the ACO sub-system with the intention of reinforcing the potentially relevant border regions in the image. This is done after observing the visual effect in Figure 3, where the substraction of a modified laplacian (Equation 5) from the gradient of an image (Equation 6) enhances border definition.

$$\hat{\nabla}^2(I) = \frac{d^2}{dx^2}I + 2\frac{d^2}{dxdy}I + \frac{d^2}{dy^2}I \tag{5}$$

$$\hat{\nabla}(I) = \frac{d}{dx}I + \frac{d}{dy}I \tag{6}$$

B. Interface Usage

Due to the considerable amount of parameters involved in the Tyagi et al. Hybrid System [18], a Graphical User Interface was produced to interactively execute the underlying algorithm, and observe the changes caused by the modification of said parameters. As part of a greater theme of study regarding band detection in DGGE images at UdeC, specific user requirements were stated for inclusion in the developed interface, the most critical of which being the option of segmenting the input image in slices prior to processing, and the possibility of altering the fuzzy rules and variables comprising the Fuzzy Inference sub-system.

A typical workflow of interface use is described as follows: From the Main Window, the user loads the input image into the program. If prefered, the Divide Into Bands Dialog can be summoned to divide the current DGGE image into band slices through user input, which can then be loaded into the software independently. Afterwards, the Adjust Parameters Dialog is accessed to set the number of ants to be used by the ACO sub-system, where this value can be automatically computed from

Fig. 4. Interface dialogs. Top-left: Main Window, top-right: Divide Into Bands, middle-left: Adjust Parameters, middle-right: Manage Fuzzy Rules, bottom-left: Manage Fuzzy Categories, bottom-right: Binarize Image.

the image's dimensions. Additional parameters from the ACO sub-system can be modified through Adjust Parameters Dialog, whereas fuzzy rules and categories from the Fuzzy Inference sub-system can be edited in the Manage Fuzzy Rules Dialog and Manage Fuzzy Categories Dialog, respectively. Optionally, the user may want to save the current model configuration as a file in order to retrieve it at a later time.

It is appropriate at this stage to begin executing the system by pressing the Start button from Main Window. A status bar will indicate the processing state of the program, and its completion will display the Binarize Image Dialog, where the resulting border map under the application of isodata thresholding, will be shown. Finally, the user will be able to alter the suggested thresholding value and save the generated image to a file from this same screen.

Figure 4 showcases the relevant screens of the interface, while Figure 5 presents the potential use cases encountered throughout the software's usage.

IV. Results

The system has been tested under different configurations and results are promising.

The system has been tested under different configurations, and its results are promising and similar to those obtained by using other mechanisms. The image shown in Figure 5, represents a test in which the proposed mechanism obtain a 79% of edges with respect to the best approach, proposed in [18]. Here a) is the original lane, b) shows the results obtained

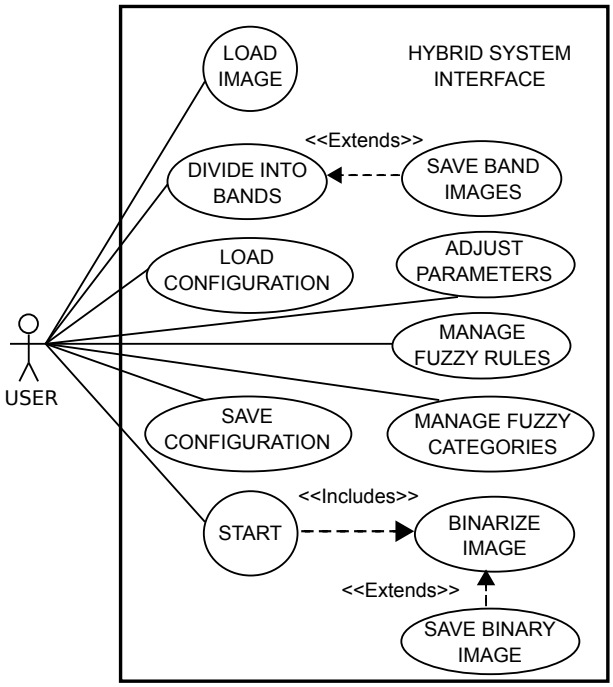


Fig. 5. Use case diagram of the hybrid system interface.

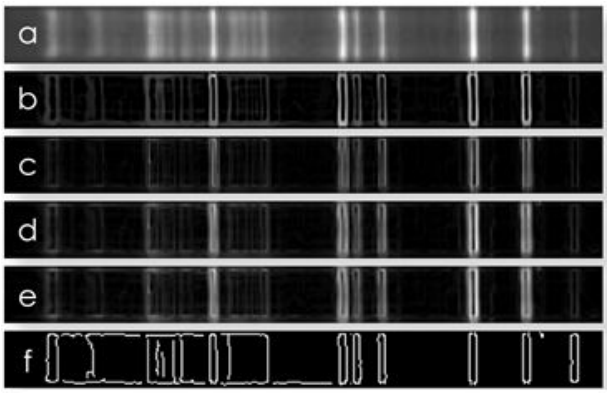


Fig. 6. Results. It takes into account different edge detection mechanisms.

with the hybrid proposal, c) is the result obtained when using Roberts, d) is the result obtained when using Prewitt, e) is the result obtained using Sobel mechanism, and f) is the result obtained with Canny. Taking into account the fact that the interface allows to manage fuzzy rules an other parameters, it is possible to obtain improved results after an adjustment of parameters and rules.

V. Conclusions

The proposed interface represents an interesting alternative to detect edges in DGGE images. The alternative pheromone matrix enhances the model in a way to focus ants' search in the regions where there is a greater probability of containing edges, while providing the added benefit of obtaining leaner borders.

On the other hand, the developed interface is a very flexible tool that permits the exploration of diverse edge detection strategies within the hybrid model, as it is designed to permit the manipulation of its intrinsic parameters, as well as to readily visualize the impact these changes have upon the general behavior of the system.

References

[1] Amza, C.G., Cicic, D.T. *Industrial Image Processing Using Fuzzy-Logic*. 25th DAAAM International Symposium on Intelligent Manufacturing and Automation, Procedia Engiineering 100 (2015), pp. 492 - 498, 2014.

[2] Anas, E. *Edge detection techniques using fuzzy logic*. International Conference on Signal Processing and Integrated Networks, pp. 169-173, 2016.

[3] Auer, C., Bachmaier, C., Brandenburg, F., Reislhuber, J. *Optical Graph Recognition*. Journal of Graph Algorithms and Applications. Vol. 17, no. 4, pp. 541 - 565, 2013.

[4] Borrajo, M.L., Baruque, B., Corchado, E., Bajo, J., Corchado, J.M. *Hybrid neural intelligent system to predict business failure in SMEs*. International Journal of Neural Systems 21(04), 277 - 296, 2011.

[5] Borresen, A.L., Hovig, E., Brogger, A. *Detection of base mutations in genomic DNA using denaturing gradient gel electrophoresis (DGGE) followed by transfer and hybridization with gene-specific probes*. Mutation Research/Fundamental nd Molecular Mechanisms of Mutagenesis 202, 77 - 83, 1988.

[6] Callan, R. *Artificial Intelligence*. Macmillan Education UK, 2003.

[7] Canny, J. *A computational approach to edge detection*. In Transactions on Pattern Analysis and Machine Intelligence, Vol. 6, pp. 679 -698, 1986.

[8] Deneubourg, J.L., Goss, S., Pasteels, J.M., Beckers, R. *Collective decisionmaking through food recruitment*. Insectes sociaux, 1990.

[9] Ziou, D., Tabbone, S., et al. *Edge detection techniques - An overview*. In Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii, Nauka/Interperiodica Publishing, Vol. 8, pp: 537-559, 1998.

[10] Dorigo, M., Stutzle, T. *Ant Colony Optimization*. The MIT Press, 2004.

[11] Dorigo, M. and Di Caro, G. *The Ant Colony Optimization Meta-Heuristic*. In New Ideas in Optimization, 1999.

[12] Izhar, ul Haq, I., Shah, K., Khan, M.T., Azam, K., Anwar, S. *Fuzzy Logic Based Edge Detection for Noisy Images*. Technical Journal, University of Engineering and Technology (UET) Taxila, Pakistan, Vol. 20(SI), No. II(S), 2015.

[13] Kaur, J., Mahajan, M. *Hybrid of Fuzzy Logic and Random Walker Method for Medical Image Segmentation*. I.J Image, Graphics and Signal Processing, Vol. 2, pp. 23 - 29, 2015.

[14] Muyzer, G,m de Waal, E.C., Uitterlinden, A.G. *Profiling of complex microbial populations by denaturig gradient gel electrophoresis analysis of polynerase chain reaction amplified genes coding for 16SrRNA*. Applied and Environmental Microbiology 59(3), 695 - 700, 1993.

[15] Nawgaje, D.D, Rajendra, Dr., Kanphade, D. *Implementation of fuzzy logic for detection of suspicious masses in mammogrammgs using DSP TMS320C6711*. International Journal of Advanced Engineering and Application, 2011.

[16] Parmee, I. *Evolutionary and Adaptive Computing Engineering Design*. Springer, 2001.

[17] Peric, N. *Fuzzy Logic and Fuzzy Set Theory Based Edge Detection Algorithm*. Serbian Journal of Electrical Engineering. Vol. 12, No. 1, pp. 109 - 116, 2015.

[18] Puntambekar, T., Sexena, P., Tanwani, P., Tyagi, V. *A Hybrid Approach to Edge Detection Using Ant Colony Optimization and Fuzzy Logic*. International Journal of Hybrid Information Technology. Vol. 5, No. 1, January 2012.

[19] Tian, J., Yu, W., Xie, S. *An ant colony optimization algorithm for image edge detection*. Evolutionary Computation 2008, CEC 2008, Hong Kong, China, September 2008.

[20] Trillas, E., Eciolaza, L. *Fuzzy Logic: An Introductory Course for Engineering Students*. Springer, 2015.

[21] Villagran, H. [*Hybridization of ACO Algorithms in Edge Detection in DGGE Images*]. Universidad de Concepcion, 2015. (In Spanish).

[22] Walad, K.P., Shetty, J. *Fuzzy Logic Based Edge Detection Using Trapezoidal and Triangular Member Function*. International Journal of Engineering Research and General Science, Vol. 3, Issue 1, January-February 2015.

[23] Zadeh, L.A. *Fuzzy Sets*. Information and Control, Vol. 8, No. 3, pp. 338 - 353, 1965.

Doble modelo de fuerzas gravitacionales para procesamiento de imagen

Cedric Marco-Detchart
Estadistica, Informatica y Matematicas
Universidad Publica de Navarra
Pamplona, Spain
cedric.marco@unavarra.com

Javier Fernandez
Estadistica, Informatica y Matematicas
Universidad Publica de Navarra
Pamplona, Spain
fcojavier.fernandez@unavarra.com

Humberto Bustince
Estadistica, Informatica y Matematicas
Universidad Publica de Navarra
Pamplona, Spain
bustince@unavarra.com

***Abstract*—En este trabajo se presenta un nuevo método de procesamiento de imagen basado en el movimiento de partículas debido a las fuerzas de atracción gravitacionales. Describimos el modelo de fuerzas gravitacionales como un método iterativo y estudiamos el efecto de sus diferentes parámetros. Probamos sus utilidad sobre dos aplicaciones clásicas de procesamiento de imagen, como son el suavizado y la segmentación. Realizamos la experimentación y el análisis de los resultados sobre imágenes en color.**

***Index Terms*—fuerza gravitacional, suavizado, segmentación, representación jerárquica**

I. Introducción

Las técnicas para abordar problemas en temas de inteligencia artificial se han inspirado considerablemente de la naturaleza. Un ejemplo clásico es el análisis de las conexiones neuronales de nuestro cerebro que ha desembocado en la teoría de las redes neuronales artificiales, cuyo estudio está actualmente en auge, por ejemplo, en problemas de clasificación [1]. Otra fuente de inspiración natural proviene de la teoría de la evolución, que ha contribuido a generar herramientas para la resolución de problemas de optimización y búsqueda de parámetros específicos en determinados problemas. Un buen ejemplo son los algoritmos genéticos, cuyo funcionamiento consiste en tomar una población de posible soluciones y hacerla evolucionar hasta llegar a una solución con el menor error posible [2]. En cuanto a la inspiración proveniente del comportamiento animal, contamos con numerosos ejemplos, como pueden ser las representaciones basadas en el funcionamiento de los enjambres [3] o las colonias de hormigas [4].

Estrechamente relacionado con el mundo natural y su comportamiento, encontramos la teoría física de la Ley de Gravitación Universal (LUG) [5], que ha servido de base para muchos investigadores a la hora de generar modelos basados en las fuerzas gravitacionales. Su objetivo es aplicar un modelo físico del mundo real a un problema matemático. Estos modelos centran su atención principalmente en temas como el clustering o los problemas de optimización.

El clustering gravitacional, presentado por Wright [6], considera los datos a procesar como partículas en un sistema, con una posición inicial y una masa, que se mueven debido a las fuerzas que ejercen entre ellas. El movimiento del sistema cesa cuando solamente queda una partícula que representa el total de los elementos. En esta misma temática una propuesta similar fue presentada por Gomez *et al*. [7] donde se describe un sistema gravitacional más elaborado. En dicho sistema se incluyen conceptos como la aceleración o la velocidad de las partículas. En cuanto a los problemas de optimización, Rashedi *et al*. presentaron el algoritmo de búsqueda gravitacional [8] que condidera la ley de gravitación y la masa de las partículas para construir un sistema multi-agente para la búsqueda de la mejor solución a un problema dado.

Respecto al procesamiento de imagen, encontramos una técnica interesante aplicada a detección de bordes presentada en [9] que considera los píxeles de una imagen como partículas y utiliza las fuerzas gravitacionales para detectar cambios de intensidad en la imagen.

Nuestro objetivo con este trabajo es aunar las propuestas presentadas en el clustering gravitacional de Wright y el detector de bordes gravitacional, aplicando un modelo de fuerzas gravitacionales para regularización de imagen y segmentación.

La aplicación que se presenta en este trabajo utiliza nuestra propuesta de manera doble. Por un lado, para obtener una imagen suavizada medimos la fuerza que ejercen cada uno de los píxeles de la imagen entre ellos. Esta fuerza representa el movimiento que se realiza en el espacio de color de manera que se vayan obteniendo regiones planas con colores homogéneos. Por otro lado, extraemos superpíxeles [10] a partir de las imágenes suavizadas y los combinamos aplicando nuestra propuesta de fuerzas gravitacionales, consiguiendo una representación jerárquica de la imagen. Dicha representación permite obtener los diferentes niveles de detalle de los objetos presentes en la imagen.

La estructura de este trabajo consta de las siguientes partes. En la Sección II exponemos algunos de los conceptos utilizados en el trabajo. La Sección III introduce el concepto de fuerza gravitacional y su aplicación al procesamiento de señal; concretamente en la Sección III-A se presenta la propuesta para suavizado de imagen y en la Sección III-B la referente a segmentación. Finalmente el estudio experimental se lleva a cabo en la Sección IV, presentando algunos resultados y conclusiones.

II. Preliminares

En esta sección recordamos algunos de los conceptos básicos relativos a procesamiento de imagen y teoría de fuerzas gravitacionales. En nuestra propuesta representamos una imagen como una función $f : D \subseteq \mathbb{R}^2 \mapsto L$, donde D representa el dominio como el producto cartesiano de dos conjuntos discretos y L el posible conjunto de valores de cada elemento (píxel) de la imagen. En concreto, $D = X \times Y = \{1, ..., w\} \times \{1, ..., h\}$, donde w representa la anchura, es decir, el número de columnas de la imagen, y h la altura, es decir, el número de filas de la imagen.

La utilización de L nos permite representar diferentes tipo sd e imagen. En el caso de imágenes binarias los píxeles pueden tomar valores en $L = \{0, 1\}$, mientras que en el caso de imágenes en escala de grises lo hacen en $L = \{0, ..., 255\}$, y para las imágenes en el espacio de color RGB el rango de posibles valores es $L = \{0, ..., 255\}^3$. Existen otros espacios de color como CIELAB, donde el rango de valores no está tan bien definido como en RGB. CIELAB consta de tres componentes (L^*ab), que representan la luminosidad (L^*), el ángulo entre los colores verde y rojo (a) y el ángulo entre el azul y el amarillo (b). Representamos una imagen que toma valores en L como $\mathbb{I}_L$.

Este trabajo se ha llevado a cabo utilizando imágenes en el espacio de color RGB para el caso del proceso de regularización y en el espacio CIELAB para la parte de clustering.

La ley de gravitación universal establece que todo objeto es atraído por otro objeto con una fuerza directamente proporcional al producto de sus masas e inversamente proporcional a la distancia que los separa. Podemos ver una representación esquemática ilustrando la fuerza entre dos partículas en la Figura 1. La fuerza existente entre dos partículas es un vector calculado mediante la Ecuación 1, donde i, j representan cada una de las partículas, m_i y m_j son sus masas y el vector $\vec{r}$ es la distancia entre ellas. Habitualmente dicha distancia es la Euclídea.

$$\vec{f}_{ij} = G \cdot \frac{m_i \cdot m_j}{\|\vec{r}\|^2} \cdot \frac{\vec{r}}{\|\vec{r}\|} \tag{1}$$

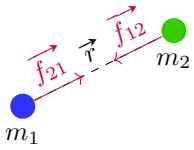


Fig. 1: Fuerza de atracción entre dos partículas

La fuerza total ejercida sobre una partícula es la suma de todas las fuerzas realizadas por el resto de partículas a su alrededor, tal como se indica en la siguiente expresión:

$$\|\vec{F_i}\| = \sum_{\substack{j \in N \\ i \neq j}} G \cdot \frac{m_i \cdot m_j}{\|\vec{r}\|^2} \cdot \frac{\vec{r}}{\|\vec{r}\|} \tag{2}$$

donde N representa todas las partículas del sistema.

III. Procesamiento gravitacional de imágenes

Desde la presentación del modelo gravitacional por parte de Wright para clustering de datos han surgido numerosas investigaciones inspiradas por este modelo. Algunas de las técnicas se han aplicado a procesamiento de imagen, en concreto al análisis de texturas [11] y a la detección de bordes [9]. Nuestra propuesta recoge la idea del modelo de Wright y aplica sus conceptos a la regularización de imágenes y a su segmentación de manera jerárquica.

A. Regularización de imagen

La regularización de imágenes sustituyendo la información de sus píxeles por medio de una función dependiente de la información de los píxeles vecinos puede llevar a una pérdida de definición de la imagen. El efecto que se obtiene es una imagen borrosa, como puede ocurrir al aplicar la media o el filtro Gaussiano. Este comportamiento puede ser válido para eliminar el ruido existente en una imagen o reducir el efecto del muestreo, pero no es apto, por ejemplo, para detección de bordes o tratamiento de imágenes por satélite. En la literatura existente encontramos técnicas que consideran el caso particular de los bordes, donde la regularización de la imagen conserva su definición. Para poder llevar a cabo una regularización de la imagen preservando sus bordes, la influencia de la regularización se debe adaptar y cambiar según las condiciones de la información local.

Existe una gran variedad de técnicas que eliminan la información innecesaria a la par que conservan aquella relativa a los bordes. Entre estas técnicas contamos con el filtrado bilateral [12], donde los valores de cada píxel son el resultado de una media ponderada dependiente de la información espacial y tonal aplicados sobre una ventana de tamaño fijo. Otra de las técnicas en este ámbito es el conocido Mean Shift [13], donde la información considerada va más allá de una ventana fija, variando su tamaño según la información local.

Adoptando el proceso de tomar la información del vecindario de un píxel, en este trabajo, imitamos el movimiento generado por la influencia de fuerzas gravitacionales existentes entre los píxeles para modificar su valor. El modelo gravitacional establece que cada partícula ejerce una influencia sobre todas las demás; con una pequeña cantidad de partículas esta norma es admisible, pero en el caso de una imagen donde el número de píxeles es muy grande, calcular cada una de las fuerzas ejercidas entre cada par de píxeles se convierte en una tarea computacionalmente prohibitiva. Debido a que la fuerza gravitacional depende de la distancia, aquellas partículas que se encuentran lejos de la posición considerada tendrán menos influencia. Aprovechando este hecho, consideramos que todas las partículas que se encuentren más allá de una determinada distancia no ejercen ninguna influencia, es decir, su fuerza es cero. La región de influencia considerada alrededor de cada píxel se establece mediante una máscara circular. Cada posición de la máscara se utiliza para saber que píxeles están ejerciendo influencia sobre la posición central.

Uno de los elementos clave del cálculo de la fuerza total (Eq. 2) es la medida de distancia considerada entre dos partícu-

las. En nuestro modelo representamos las partículas mediante un vector n-dimensional. Por ejemplo, para una imagen RGB, el píxel de la posición p sería $\mathbb{I}(p) = [x, y, r, g, b]$. En el caso de imágenes en el espacio de color RGB combinamos la posición espacial del píxel con cada una de las componentes de color. Mientras que para los descriptores espaciales, $[x, y]$, no se conoce a priori su rango de valores ya que depende del tamaño de la imagen, para la información relativa al color, $[r, g, b]$, los posibles valores se encuentran en L. El principal problema existente con esta representación es la disimilitud entre los rangos de posibles valores de cada uno de los componentes.

Si consideramos el descriptor completo, es decir, los cinco componentes que representan la posición y la información relativa al color, puede darse una inconsistencia en el cálculo de la distancia. Además, si normalizamos los valores de la intensidad de color para llevarlos al rango $[0, 1]$, la diferencia de rango entre la información espacial y de color es todavía mayor. Hacer uso de la distancia Euclídea directamente con el vector 5-dimensional resultaría en una distancia influenciada principalmente por la información espacial.

Partiendo de la base de la distancia Euclídea, separamos su cálculo en una parte espacial y otra tonal. Así podemos añadir un peso al cálculo de la distancia que nos permite controlar la influencia de cada una de las partes. Con esta modificación el cálculo de la distancia se realiza mediante la siguiente expresión:

$$r_s = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$
$$r_c = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}$$
$$r = r_s + r_c \cdot \omega_c$$

El factor de influencia, ω_c, nos permite controlar la importancia asignada o bien a la distancia espacial o a la distancia en color. Tal como se observa en la experimentación con este parámetro somos capaces además de controlar el nivel de borrosidad aplicado al final del proceso. Para poder obtener resultados admisibles un cuanto a suavizado sin perder definición de bordes, los experimentos realizados sugieren que el valor de ω_c debe tomare en el rango $[1, 100]$.

En un primer paso, dada una máscara m de radio $\mathfrak{r} = 7$ calculamos una imagen de fuerzas, representada como $\mathbb{I}_F$. Cada una de las posiciones p de dicha imagen en cada uno de los canales (RGB) indica la fuerza total (Eq. 2) que actúa sobre esa posición. Esto significa que en cada posición tenemos el valor de intensidad que debe variar el píxel de esa posición. La fuerza que se ejerce sobre cada una de las partículas se calcula considerando que su masa es igual a uno, ya que cada partícula representa un píxel.

El principal problema que surge con el cálculo de la fuerza es que el valor final de la información de color se mantenga en el rango L. Si la fuerza obtenida es demasiado fuerte, el resultado de aplicarla sobre la intensidad de color actual del píxel correspondiente puede resultar en un valor fuera de rango y perder sentido. La solución aportada es hacer uso de la constante gravitacional G presente en la Ecuación 1. Esta constante nos permite adaptar la fuerza final y hacer que tenga mayor o menor influencia. En este sentido, tomar valores de G altos hace que las partículas del sistema gravitacional varíen con mayor intensidad de manera que se obtiene una imagen suavizada en un menor lapso de tiempo. Por el contrario, con valores bajos de G la variación será menor y el resultado final del suavizado tardará mas tiempo.

Entre el número de iteraciones máximo y los dos parámetros, ω_c y G, de nuestro modelo tenemos un amplio rango de posibilidades para poder controlar el nivel de suavizado de las imágenes. En el caso de G elegimos valores comprendidos entre $(0, 0.1]$. Además, debido a que estamos imitando un sistema gravitacional, la variación de intensidad debe ser baja para representar el movimiento constante de las partículas. Si $G \approx 0$, la fuerza ejercida sobre cada píxel será casi imperceptible, mientras que si tomamos $G = 0.1$ la intensidad de los píxeles variarán en un orden máximo de 10^{-3}. Estos órdenes son consecuentes con el factor de color aplicado en el cálculo de la distancia. Una vez calculada la fuerza total sobre cada posición se aplica sobre la imagen de manera iterativa:

$$\mathbb{I}_L(p)_{t+1} = \mathbb{I}_L(p)_t + \mathbb{I}_F(p)_t \tag{3}$$

Para el cálculo de los nuevos valores en cada una de las posiciones de la imagen suavizada solamente consideramos la fuerza ejercida sobre la información relativa a la intensidad de color. De este modo mantenemos la estructura matricial de los píxeles y obtenemos una imagen regularizada en el espacio de color.

B. Segmentación de imagen

La segmentación de imagen es una de las tareas más importantes en visión por computador. Es una de las tareas más complejas que ha sido y sigue siendo estudiada en profundidad [14]. El principal reto de la segmentación de imagen radica en el hecho de que no existe una única solución. Incluso para el caso de los seres humanos, que son capaces de identificar rápidamente objetos en una imagen, la definición de un objeto o de sus partes puede ser múltiple y no presentar una forma clara. Esta problemática puede verse reflejada en las imágenes proporcionadas por los expertos del conjunto de datos de Berkeley Segmentation Dataset (BSDS) [15], donde cada imagen cuenta con hasta cinco posibles soluciones propuestas por seres humanos. En este sentido, aunque las personas son capaces de percibir muchos patrones y asociarlos con objetos, el hecho de definir una partición de una imagen es una tarea compleja.

Además, no todos los objetos de una imagen tienen la misma importancia, se deben tener en cuenta los diferentes niveles de detalle que pueden existir. Los estudios llevados a cabo sobre la percepción humana establecen que las personas son capaces de definir objetos a diferentes niveles de detalle [16], asociando partes de la imagen a través de propiedades que las caracterizan, como por ejemplo, el color o la forma. Para poder gestionar los diferentes niveles de detalle en imágenes, e imitar la percepción humana, los investigadores

se han centrado en la segmentación jerárquica o la fusión de regiones [17]. Los diferentes niveles de una jerarquía se pueden representar mediante lo que se conoce como un mapa de contornos ultra métrico (UCM) [18],

En nuestra propuesta generamos una partición inicial de la imagen formada por regiones primarias conocidas como superpíxeles. Esta imagen es una sobre-segmentación, lo que significa que hay más particiones que objetos reales en la imagen. Generamos los superpíxeles iniciales mediante la aplicación de la transformada watershed [19] sobre una imagen de magnitud de gradiente de una imagen pre-procesada con el algoritmo de suavizado gravitacional. Para limitar el numero de superpíxeles extraídos eliminamos todos aquellos valores de la magnitud del gradiente que se encuentran por debajo de la mediana de todos los mínimos regionales [20]. A partir de los superpíxeles iniciales, los vamos fusionando de manera progresiva hasta que únicamente quede uno solo, que representa el contenido de toda la imagen.

Contrariamente al proceso de suavizado, cuando trabajamos con superpíxeles, la masa de cada partícula depende del tamaño del superpíxel, es decir, del número de píxeles que contiene esa región. Para ajustar los valores de las masas a un espacio mas reducido les aplicamos una escala logarítmica. Además, la distancia que utilizamos en esta fase de agrupación de regiones varía con respecto a la sección anterior; en este caso no medimos la distancia entre elementos individuales. El cálculo de la distancia del color permanece intacta, medida en este caso entre el color medio de los superpíxeles, mientras que la distancia espacial se realiza utilizando la siguiente expresión:

$$D_s = 1 + d\{A, B\}$$

donde d es la mínima distancia Euclídea entre los superpíxeles A y B ; obtenemos así la distancia ente la pareja de píxeles mas cercanos de las dos regiones. Posteriormente combinamos la distancia entre la información del color y la espacial mediante el factor ω_c.

IV. Estudio experimental

El algoritmo gravitacional propuesto en este trabajo en la Sección III se ilustra a través del siguiente experimento, aplicado a suavizado y segmentación de imagen.

Por un lado, examinamos la influencia que el número de iteraciones y el parámetro ω_c tienen sobre el suavizado gravitacional. Por otro lado, dadas las diferentes imágenes suavizadas evaluamos como afecta el suavizado en el proceso de segmentación y fusión jerárquica. Evaluamos el algoritmo propuesto con un máximo de 300 iteraciones y tomando una serie de instantáneas cada cierto instante de tiempo $t = \{10, 40, 80, 100, 140, 220, 260, 300\}$. En el caso del proceso de segmentación utilizamos el operador de Canny [21] con $\sigma = 2$ para obtener la magnitud del gradiente de la imagen. Con el resultado obtenido construimos un mapa de superpíxeles mediante la transformada watershed.

Para la realización de los experimentos hemos utilizado 200 imágenes del conjunto de datos de test del Berkeley Segmentation Dataset (BSDS500) [15]. Para cada UCM obtenida en cada instante de tiempo t aplicamos un umbral $th = \{0.17, 0.33, 0.50, 0.67, 0.83\}$ para obtener una imagen de bordes binaria. Dicha imagen de bordes se evalúa como un proceso de correspondencia con las imágenes de los expertos proporcionadas por el conjunto de datos. Este proceso de correspondencia se lleva a cabo mediante la técnica presentada por Estrada y Jepson [22] calculando las siguientes medidas de precisión ($Prec$) y exhaustividad (Rec):

$$Prec = \frac{TP}{TP + FP}, \; Rec = \frac{TP}{TP + FN},$$
$$F_\alpha = \frac{Prec \cdot Rec}{\alpha \cdot Prec + (1 - \alpha) \cdot Rec}.$$

Para el cálculo de F_α utilizamos $\alpha = 0.5$ tal como se indica en [23].

Como se muestra en la Figura 2 el paso del tiempo afecta el proceso de suavizado. A medida que las iteraciones van creciendo, van apareciendo zonas planas en aquellas regiones donde los colores son similares. Se preservan los bordes de los objetos hasta que las regiones son completamente homogéneas Cuando una región acaba siendo totalmente plana, los diferentes colores cercanos comienzan a mezclarse resultando en un suavizado de los bordes y por consiguiente en una pérdida de definición de los mismos.

Podemos observar en la parte superior que con un valor bajo de $\omega_c = 40$ el efecto borroso es más pronunciado. Si nos centramos en los dos peces cerca de los corales, podemos ver como la parte blanca de la aleta se va difuminando progresivamente hasta que acaba desapareciendo fusionándose con la parte más oscura del mar. En la parte inferior de la figura, los dos peces permanecen bien definidos a lo largo de todo el proceso iterativo, conservando sus bordes.

La influencia del proceso de suavizado se puede observar claramente en la Figura 3. La segmentación inicial contiene una menor cantidad de superpíxeles conforme el proceso de suavizado homogeneiza los colores de la imagen. Esta reducción en el número de superpíxeles inicial tiene un efecto directo en la construcción de la jerarquía ya que las pequeñas regiones aisladas desaparecen, quedando absorbidas por regiones contiguas al inicio del proceso.

En cuanto a la evaluación cuantitativa, en la Figura 4, observamos como el efecto de suavizado generado por el valor bajo de $\omega_c = 40$ (Figure 4a), afecta significativamente al proceso de segmentación. En las iteraciones iniciales, hasta $t = 80$, el valor de $F_{0.5}$ se mantiene estable en todos los umbrales aplicados. Por un lado, observamos como en las iteraciones iniciales los mejores valores se obtienen con el segundo umbral. Esto indica que con un menor número de iteraciones necesitamos un umbral mayor para eliminar las regiones de menor tamaño que no son aptas para definir los contornos de la imagen. Por otro lado, en las iteraciones finales obtenemos el valor $F_{0.5}$ más alto con el menor umbral. Esto significa que el efecto de suavizado ha provocado borrosidad y los diferentes colores se han fusionado por lo que la distancia entre superpíxeles es menor.

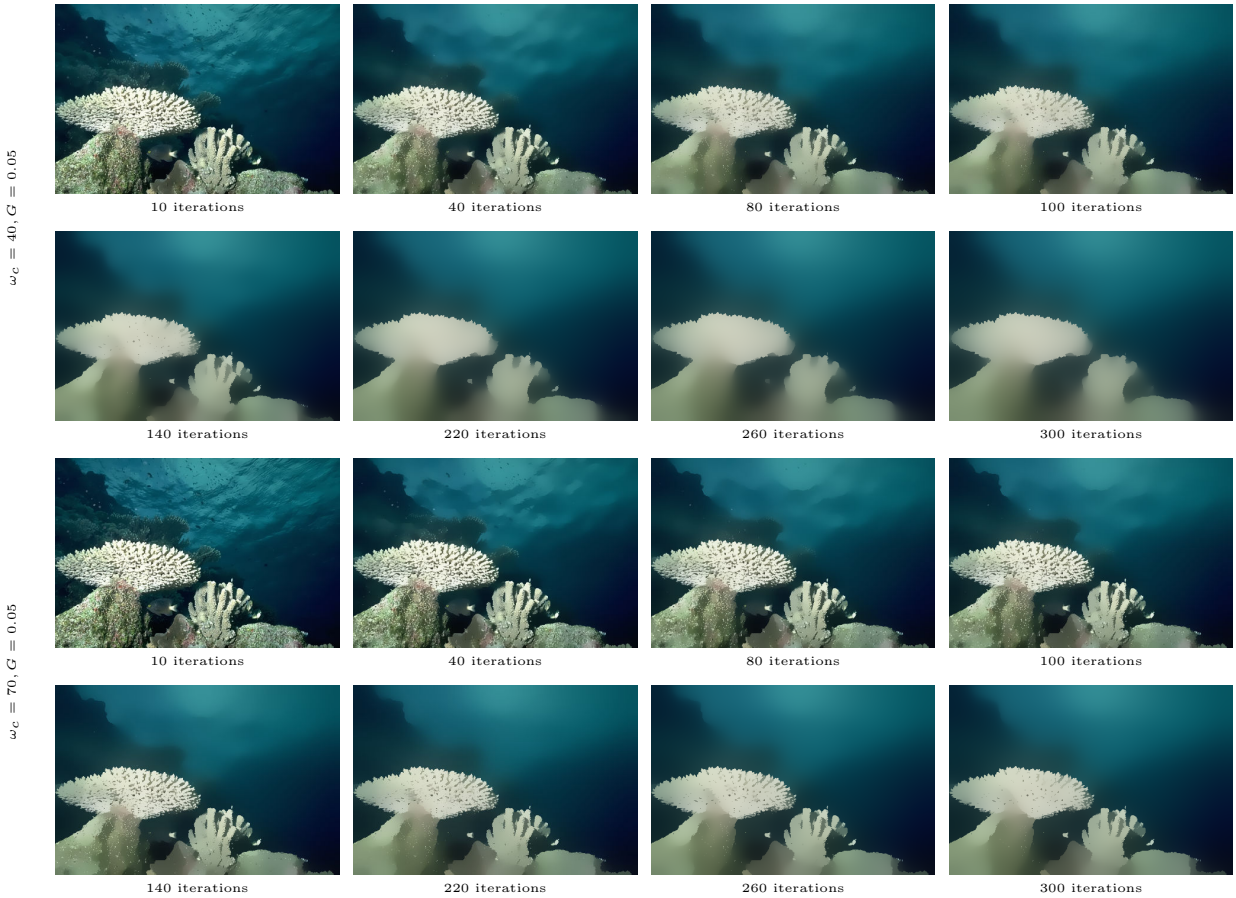
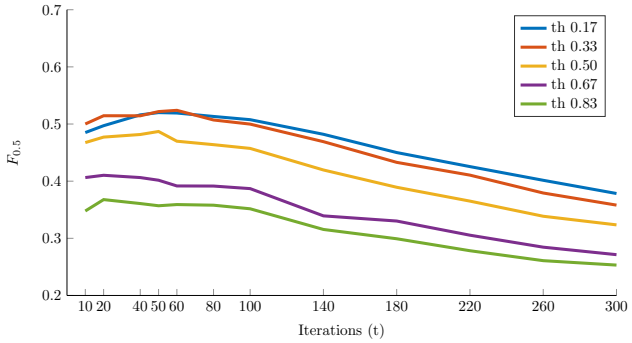
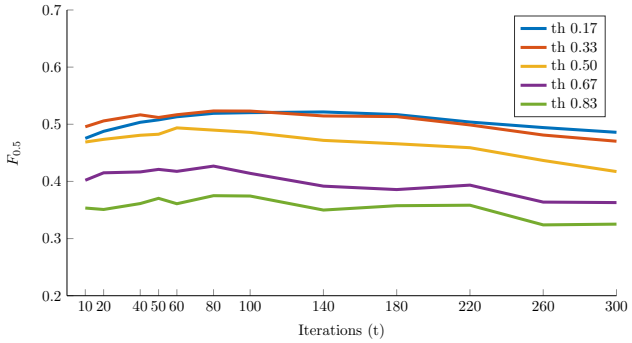


Fig. 2: Resultados obtenidos con las aplicación de nuestro algoritmo de suavizado gravitacional sobre la imagen 101027 del BSDS500 en diferentes instantes de tiempo. Utilizamos el factor de color $\omega_c \in \{40, 70\}$, la constante $G = 0.05$ y ejecutamos el método con 300 iteraciones.

Fig. 3: Imágenes de superpíxeles obtenidas en diferentes tiempos t del suavizado gravitacional sobre la imagen 118035 del BSDS500. Utilizamos el factor de color $\omega_c = 70$ y $G = 0.05$.



(a) $\omega_c = 40$



(b) $\omega_c = 70$

Fig. 4: Medida de F_α sobre 300 iteraciones con $G = 0.05$ aplicando diferentes umbrales sobre las imágenes UCM obtenidas en el proceso de segmentación jerárquica.

Al observar el proceso, pero con un valor de $\omega_c = 70$ en la Figura 4b vemos como los valore de $F_{0.5}$ se mantienen más estables a lo largo del proceso. Este comportamiento confirma el hecho de que el efecto de borrosidad es menos pronunciado y se conserva una buena definición de los bordes. El mejor resultado para este valor de ω_c se obtiene en el punto intermedio del proceso, en la iteración $t = 140$, A partir de este punto los resultados comienzan a decaer, pero la disminución es menos pronunciada que en el caso de $\omega_c = 40$.

V. Conclusiones

Hemos presentado un nuevo método para obtener imágenes suavizadas, mediante un modelo físico basado en las fuerzas gravitacionales de atracción entre partículas. Hemos introducido los conceptos matemáticos necesarios para el uso del modelo expuesto en problemas relacionados con información proveniente de imágenes. Hemos presentado dos posibles aplicaciones de nuestro modelo junto con los resultados obtenidos, probando diferentes parámetros y demostrando su utilidad en el campo de la segmentación de imágenes. Como experimento inicial los resultados obtenidos son prometedores, pero debemos realizar mas experimentos, con un estudio más profundo de los diferentes parámetros para confirmar los resultados presentados. Deberemos estudiar otras medidas de distancia, así como poner a prueba nuestro método con otros conjuntos de datos y espacios de color.

Acknowledgment

Este trabajo ha sido parcialmente financiado por el proyecto TIN2016-77356-P y por el Servicio de Investigación Universidad Publica de Navarra.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *ImageNet Classif. with Deep Convolutional Neural Networks*, pp. 1097–1105, 2012.

[2] J. Benediktsson and P. Swain, "Consensus theoretic classification methods," *IEEE Trans. Syst. Man. Cybern.*, vol. 22, no. 4, pp. 688–704, 1992.

[3] M. Wachowiak, R. Smolikova, Y. Zheng, J. Zurada, and A. Elmaghraby, "An Approach to Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 289–301, jun 2004.

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[5] I. Newton, "Philosophiae Naturalis Principia Mathematica," *Pan*, p. 510, 1687.

[6] W. E. Wright, "Gravitational clustering," *Pattern Recognit.*, vol. 9, no. 3, pp. 151–166, 1977.

[7] J. Gomez, D. Dasgupta, and O. Nasraoui, "A New Gravitational Clustering Algorithm," in *Proc. 2003 SIAM Int. Conf. Data Min.* Philadelphia, PA: Society for Industrial and Applied Mathematics, may 2003, pp. 83–94.

[8] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A Gravitational Search Algorithm," *Inf. Sci. (Ny).*, vol. 179, no. 13, pp. 2232–2248, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2009.03.004

[9] C. Lopez-Molina, H. Bustince, J. Fernandez, P. Couto, and B. De Baets, "A gravitational approach to edge detection based on triangular norms," *Pattern Recognit.*, vol. 43, no. 11, pp. 3730–3741, 2010.

[10] X. Ren and J. Malik, "Learning a classification model for segmentation," *Proc. Ninth IEEE Int. Conf. Comput. Vis.*, vol. 1, no. c, pp. 10–17 vol.1, 2003.

[11] J. J. De Mesquita Sá, A. R. Backes, and P. C. Cortez, "A simplified gravitational model for texture analysis," *J. Math. Imaging Vis.*, vol. 47, no. 1-2, pp. 70–78, 2013.

[12] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images," *Int. Conf. Comput. Vis.*, pp. 839–846, 1998.

[13] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[14] H. Zhu, F. Meng, J. Cai, and S. Lu, "Beyond pixels: A comprehensive survey from bottom-up to semantic image segmentation and cosegmentation," pp. 12–27, 2016.

[15] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour Detection and Hierarchical Image Segmentation," *Tpami*, vol. 33, no. 5, pp. 898–916, 2011.

[16] D. Marr, "Vision," *book*, 1982.

[17] F. Calderero and F. Marques, "Region merging techniques using information theory statistical measures." *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1567–86, 2010.

[18] P. Arbeláez, "Boundary extraction in natural images using ultrametric contour maps," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2006, 2006.

[19] S. Beucher, "The Watershed Transformation Applied to Image Segmentation," in *Proc. 10th Pfefferkorn Conf. Signal Image Process. Microsc. Microanal.*, 1992, pp. 299–314.

[20] P. Soille, *Morphological Image Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[21] J. F. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[22] F. J. Estrada and A. D. Jepson, "Benchmarking Image Segmentation Algorithms," *Int. J. Comput. Vis.*, vol. 85, no. 2, pp. 167–181, nov 2009.

[23] C. Lopez-Molina, B. De Baets, and H. Bustince, "Quantitative error measures for edge detection," *Pattern Recognit.*, vol. 46, no. 4, pp. 1125–1139, apr 2013.