ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

Cognitive modeling for robotics in manufacturing

TOWARDS SEQUENTIAL PROBLEM SOLVING IN ACT-R: A CASE STUDY OF TANGRAM

CANDIDATE
Giacomo Zamprogno

SUPERVISOR
Prof. Giuseppe Di Pellegrino

CO-SUPERVISOR

co-supervisor

Academic year 2021-2022

Session 1st

+retrieval>

ISA dedication

NAME tbd

REASON tbd

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Where I describe cognitive modelling and its applications, and the specific case study of the tangram in the context manufacturing and tutoring

What is cognitive modelling? Why how does it relate to AI and why is it of interest at airbus? MAYBE the reserch group at airbus The idea of the project

1.1 The Tangram

The tangram is an ancient Chinese puzzle in which seven pieces, also called *tans* are obtained from an original square.

The most common tans, also used in this work, consist of 5 square triangles (2 small, 1 medium and 2 large), 1 square and 1 parallelogram, their dimension is shown in [reference figure].

Usually players are presented with an homogeneous silhouette, likely in the shape of some stylized figure, and are tasked with reproducing such pattern by using all the tans, without overlap.

Besides the interest among puzzle-solvers, a number of works has been studying its applications in the teaching fields, where its nature as a fairly complex game can help children to develop geometric and communicative skills[citation to some papers might be needed here].

In the context of cognitive modelling, the tangram can be seen as an abstraction for a set of different sequential problem solving tasks. The fact that the field is still at the early stages of development, studying a simpler puzzle and how humans approach its solution might provide initial insights about various types of assembling processes, in an attempt to create machines more capable to interpret and adapt to human actions in an explainable and rule-based way.

Chapter 2

Related Works

Where I quickly go throught the available Literature, describe ACT-R and its functioning and analyze the various approaches for tangram solving

2.1 ACT-R

2.2 Cogntive modelling of puzzles

Despite their nature and potential as an abstraction for more sophisticated sequential problem solving tasks, the applicaions of cognitive modelling to puzzles are still at an early stage.

Rosenberg et al. [4] coupled cognitive architectures and the tangram puzzle in order to model the curiosity aspect of a social robot, but the actual solution of the puzzle was implemented with a connectionist approach and the cognitive aspect was focused on the social interaction and artificial curioisy modelling.

Gentile and Lieto [2] instead used ACT-R in order to model the role of mental rotation applied at the task of the Tetris™ video-game, based on the previous work of Shepard and Metzler[5], providing introductory results and a functioning model for the mental rotation process.

2.3 The Tangram

As mentioned, cognitive-modelling specific literature regarding the tangram is extremely limited. Nonetheless, previous works in the field of computer science and cognitive psychology can be seen as an interesting starting baseline for the modelling process.

Among the computational approaches, Deutsch and Hayes [1] originally suggested a heuristic solution based on extending the lines inside the silhouette and matching the edges of the tans with the prolonged lines; while their *direct-match* and *2.5-3.5* rules might resemble what will be later defined as *perfect fit*, their fully geometrical algorithm, and especially the evaluations on the extendeded lines are unlikely to have cognitive plausibility.

During the advent of machine learning, Oflazer [3] used a Boltzmann Machine, where each neuron represented a possible positioning of one piece, and was fed with excitatory input from the outline constraints and inhibitory input from the conflicting configurations. The main limit of the approach is that the authors only tackle *grid tangrams*, where each corner of a placed tan must be coincident with a point in an equally spaced grid. This in turn largely limits the allowed rotations for the pieces and thus its applicability, even at a knwoledge represenation level, when dealing with less constrained puzzles as the irrational length of edges would be impossible to tackle using the grid.

Chapter 3

Experimental scenario

In collaboration with the Technical University of Berlin [shall I actually mention it?] an experiment was developed and performed in order to obtain the training and test sets.

Each participant was presented with a virtual implmentation of the tanrgam game, including 4 puzzles [cite figure]. After an inital explanation of the controls, each participant was required tackle the solution of the puzzles, which were always shwon in the same order. There was no time limit, and at any moment the *NEXT* button was available, so that the player could give up on the specific tangram and move to the next one. It must be noted that while backtracking was possible during the solution of the single task, once the button was pressed the solution was submitted and no option to go back was available.

Between each problem, the participants were asked to complete a NASA evaluation form, and at the end of the trial they provided a general feedback.

In addition to this, the screen recording and application logs, including data regarding times, piece action types (rotation, movement) and piece positioning were stored for the analysis. During the first trial set, 31 participants (17 males, 14 females) took part to the experiment; they all had an academic background, a large majority of them being from engineering or human factors studies.

These first players composed the training set.

Chapter 4

Data Analysis and Hypothesis

Where I qualitatively and quantitatively analyze the available data, provide figures and introduce the leading Hypothesis that the model will try to explain

Chapter 5

Model description

In principle, it is possible to create and run an experimental setting within the ACT-R context. In the specific case of the project, however, the available data were strictly dependant on the implementation of the experimental window: the coordinates for the placements of the tans were expressed with respect to the Turtle window, and the tans themselves were defined as shapes within the framework. Reproducing the setting purely in ACT-R would likely have caused loss of precision in the shapes and coordinates definition, besides an additional overhead due to the lack of established methods for image manipulation: the creation an interfacing system was then preferred.
As a consequence, the final model consists of multiple sub-components and a coordinator, which handles the communication between the experimental window, the reasoning agent, and a visual system simulation.

5.1 Experiment window

The experimental window is mostly derived by the original code. No interactive features are required: as there is no modelling of the motor system from the agent, it is enough to provide an updating functionality that respects the state evolution after a certain action.

As the Turtle library used in the original code is not meant for external interaction, the simplest way to provide the state information to the visual system is to capture the screen, crop the interested window, and save it for further loading. This is the main additional functionality added to the module, besides a general clean-up and the removal of the interaction features.

5.2 Visual system

The agent's visual system was implemented in order to be "cognitively inspired" from the hierarchical structure of the human's visual system. Given the strictly geometrical nature of the puzzle, and the fact that *second layer neurons* [need the proper name and citation] fire when matching certain oriented lines in the visual field, the algorithm simulates an hypothetical higher level construct able to identify certain line patterns representing the shapes. This is done via a pattern matching function applied on the edges of the current state and the template, using the sum of squared differences as similarity function. Considering the binary nature of the images, a sum of absolute differences or a custom function might have been possible alternatives, but due to the amount of templates to match (for each shape and for each of its available rotations) the faster opencv implementation was preferred, even though limited in the similarity functions options.
For each template, 7 candidate placements are extracted. These are then filtered in two successive steps. First, it is checked whether the placement would intersect with other pieces currently placed, which can happen as the similarity function on edges might still tolerate limited intersection of corners. This is done by imposing a threshold on the following similarity function:

$$\sum_{x,y}((part \ \wedge \ template) \oplus template) \tag{5.1}$$

Where $\wedge$ denotes bitwise-and, $\oplus$ bitwise-xor, and *part* is the part of the target image in the candidate bounding box.

The second screening checks for feasibility: the template matching method applied aims to simulate a possible cognitive mechanism, but is in no way plausible per-se and might find patterns that are unseen or hardly seen by the humans. As a consequence, only the placements that have some representation in the training data at the current phase of the solution are kept. This also tackles an additional aspect: once a landmark is found its strength must be defined, which in turn will determine the baseline activation of the landmark chunk itself in the model. While a possible solution would be basing the strength on the similarity value, this would not have any cognitive foundation. On the other hand, the data provide a clear indicator of its likelihood: if a large number of participants acted on the landmark in the current phase of the game, it should in turn mean that such landmark should be a strong landmark. The frequency count is thus used not only for screening, but also for defining the baseline activation of the chunk once loaded into the declarative memory.

As the human's visual (working?) memory is limited [cite maybe 7+-2, or something more relevant], only the 6 strongest landmarks will define the imaginal buffer, with the rest being merely loaded into the declarative memory with their respective strength.

A final task performed by the visual system, is to recognize the presence of problematic regions (UNF-REGIONS): in principle, such regions are parts of the silhouette in which no tan can be placed respecting the rules. They cannot be matched directly as they come in various different shapes and sizes, but a property of the task can be exploited to identify some of them: the silhouette area must eventually be fully covered and all available pieces used. Considering this, if at any point in time no available placement is found for a piece,it means that its area is split between two or more separate regions that cannot accommodate it. In such cases, the presence of a problematic region is found and the last action is marked as uncertain.

5.3 Coordinator

The coordinator is the main python application, providing the interfacing features between the other modules. It initializes the various sub-components, keeps an internal representation of the puzzle state and updates it accordance to the chosen ACT-R action.

5.3.1 State representation

The tangram puzzle currently being solved is represented by the eponymous class *Puzzle*. Its fields are mainly used for listing the free tans, the currently used tans, their respective positioning and the landmarks they are involved in. While in principle the definition of the landmark already includes the involved piece, the presence of multiple tans of the same type requires such distinction, in order to avoid acting on already placed landmarks. A given action is thus defined as a tuple (Piece, Landmark). A list of the overall sequence of actions is finally stored for the performance evaluation.

5.3.2 Updating actions

While the ACT-R module will provide the reasoning and eventually choose the action performed at each step, the implementation of such action is delegated to the coordinator. The main methods that provide such functionalities are described in the following:

- **update**: the update method represents the main operation for the task solution. Accepting a landmark definition (piece type, position, orientation) it picks the first available piece corresponding to such type, creates the (Piece, Landmark) match, and updates the state accordingly.

- **region_backtrack**: this method ideally represents the attempt for backtracking when a region that cannot fit any available piece is identified. Humans (and thus the agent) do not necessarily act on such problematic

regions in the moment they produce them, but only once they are noticed.

In order to simulate such behaviour, the visual system will flag such occurrence as soon as it appears, by creating an ”UNF-REGION” landmark, but the method will be called only once the agent actually matches it via firing the backtracking production. All actions performed while such a landmark exists will be flagged as potentially problematic, and every time the method is called, the first such action will be backtracked, possibly clearing the queue if the issue is not present anymore.

- **piece_backtrack**: backtracking can happen even if no problematic region is matched: participants can realize that a current placement is leading to no solution, or fail to identify a possible landmark. Both of these cases are represented by a retrieval failure, which in turn triggers the piece_backtrack method.

  In this case no information about problematic placements is available, so the decision on which action to backtrack is based on the landmark strength, as determined by the data. For each of the active landmark the function extracts its frequency count at the current game state, and the weakest one is chosen for backtracking.

5.4 ACT-R model

The ACT-R model is tasked with choosing the next action to take at any given position, and it is mostly based on the baseline activation and spreading activation mechanisms; three modules are involved in the process: the imaginal module, the declarative module and the retrieval module.

As previously described, the visual system extracts the current landmarks, adds them to the declarative memory according to their strength, and subsequently loads the 6 strongest into the imaginal buffer. In this context, the imaginal buffer represents the ”most noticeable” landmarks, and helps their

retrieval by the spreading activation mechanism.
A description of the main production rules follow:

- **retrieve-landmark**: as the name suggests, this production calls for a retrieval of any given landmark. It matches when there are still pieces available and at least one noticed landmark is in the imaginal buffer.It poses no restrictions on the retrieval request, other than asking for a landmark chunk which has not been recently retrieved.
  The retrieval process is then guided by three components: the baseline activation, which is provided at chunk definition (by the visual system) and depends on the landmark strength, the spreading activation, depending on the landmark chunks present in the imaginal buffer, and a noise component, the distribution of which is an hyperparameter of the model.

- **act-and-update**: this production is triggered whenever the retrieval from **retrieve-landmark** is successful. Its task is to extract the relevant fields from the landmark chunk and to provide them to the coordinator, by calling the **update** method.

- **fail-to-retrieve**: the experiments additionally show that a participant might fail to retrieve certain landmarks even if they are visible and representing a near ”perfect-match”. In order to simulate such behaviour, a retrieval threshold value is introduced: if, due to noise or weakness of activation, no landmark's activation is above the threshold a buffer failure will happen in the retrieval request. Such failure is matched by the production **fail-to-retrieve**, its purpose being to trigger the **piece_backtrack** procedure.

- **notice-problem**: this production is in direct competition with **retrieve-landmark** whenever the special landmark *UNF-REG* is present in the

imaginal buffer. This represents the participant noticing a problematic region in the puzzle and thus starting the backtracking process of solving such region.

- **solve-problem**: this production directy follows **notice-problem** and, in a similar fashion to **fail-to-retrieve**, its task is to interact with the coordinator in order to call the correct backtracking method.

The interaction between the two backtracking procedures should be noted: in some cases, a given incorrect placement of a piece does not create any recognizable unfeasible-region, but will cause all the following actions to be problematic. This will not be noticed by any of the two individual productions, but it will eventually be possible to backtrack by their combination. Due to the constraint on *:recently-retrieved nil*, after various iterations of the **region_backtracking** with no solution, no new landmark will be available. At this point the **piece_backtracking** method will be triggered, which will remove the weakest landmark. At later phases of the solution, the problematic landmarks will be less frequent, as more participants will actually have solved the puzzle, and its strength in the model will thus be lowered.

Chapter 6

Results and discussion

Where I compare the model to the expected data and try to discuss whether the hypothesis are funded and whether there are possible applications

Bibliography

[1] E. S. Deutsch and K. C. H. Jr. “a heuristic solution to the tangram puzzle”. *Machine Intelligence*, 7, 1972.

[2] M. Gentile and A. Lieto. The role of mental rotation in tetristm gameplay: an act-r computational cognitive model. *Cognitive Systems Research*, 73:1–11, 2022. ISSN: 1389-0417. DOI: https://doi.org/10.1016/j.cogsys.2021.12.005. URL: https://www.sciencedirect.com/science/article/pii/S1389041721000991.

[3] K. Oflazer. Solving tangram puzzles: a connectionist approach. *International Journal of Intelligent Systems*, 8, 1993.

[4] M. Rosenberg, H. W. Park, R. Rosenberg-Kima, S. Ali, A. K. Ostrowski, C. Breazeal, and G. Gordon. Expressive cognitive architecture for a curious social robot. *ACM Trans. Interact. Intell. Syst.*, 11(2), 2021. ISSN: 2160-6455. DOI: 10.1145/3451531. URL: https://doi.org/10.1145/3451531.

[5] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971.

Acknowledgements