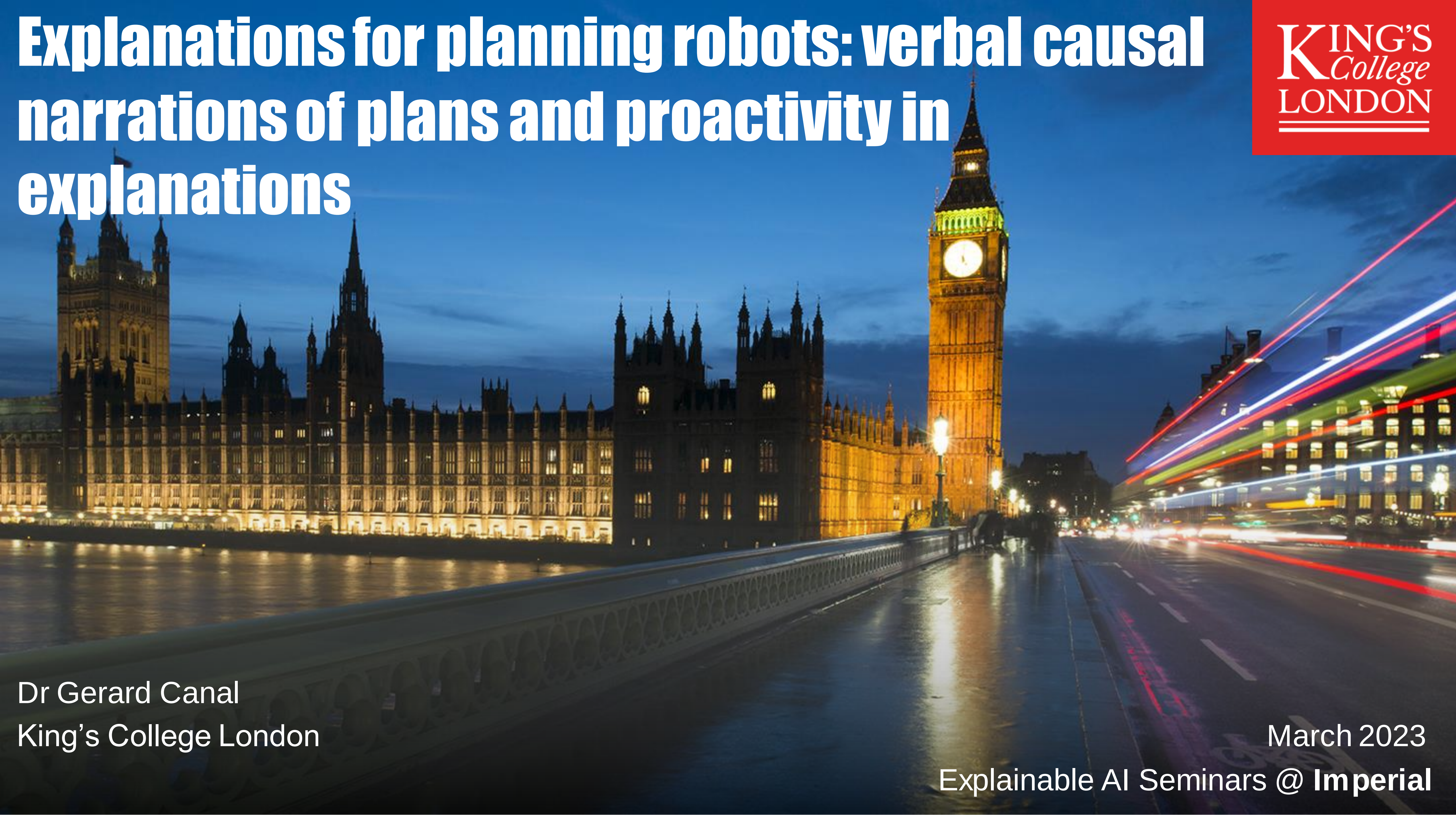
Explanations for planning robots: verbal causal narrations of plans and proactivity in explanations



Dr Gerard Canal
King's College London

March 2023

Explainable AI Seminars @ **Imperial**

About me...

- BSc in Computer Science (2013) and MSc in Artificial Intelligence (2015)
  - Universitat Politècnica de Catalunya (BSc, MSc), Universitat de Barcelona, Universitat Rovira i Virgili (MSc)
- PhD in Robotics, Automatics, and Vision (2020)
  - Universitat Politècnica de Catalunya
- Research Associate / UK IC Postdoctoral Research Fellow (2020/2021)
  - King's College London
- Lecturer in Autonomous Systems (2022)
  - King's College London

- Research Interests in:
  - Assistive Robotics
  - Planning/Decision making for robotics
  - **Explainability**

About the talk…

- Verbalisation of task plans with causal information

  Gerard Canal, Senka Krivić, Paul Luff, and Andrew Coles. “PlanVerb: Domain-Independent Verbalization and Summary of Task Plans”. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 36(9). June 2022, pp. 9698–9706.

- Detection of explanation triggers through gaze analysis

  Lennart Wachowiak, Peter Tisnikar, Gerard Canal, Andrew Coles, Matteo Leonetti, and Oya Celiktutan. “Analysing Eye Gaze Patterns During Confusion and Errors in Human–Agent Collaborations”. In: 31st IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN). 2022.

Verbalising Task Plans with Causal Justifications

Gerard Canal, Senka Krivić, Paul Luff, and Andrew Coles. “PlanVerb: Domain-Independent Verbalization and Summary of Task Plans”. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 36(9). June 2022, pp. 9698–9706.

Introduction and motivation

- Task plans may not be easy to understand by lay users.

```
0.000: (goto_waypoint robot_assistant wp3 wp0) [14.000]
0.000: (goto_waypoint robot_helper wp1 wp2) [7.000]
7.001: (goto_waypoint robot_helper wp2 wp5) [19.000]
14.001: (scan_place robot_assistant wp0) [1.000]
15.002: (goto_waypoint robot_assistant wp0 wp16) [18.000]
26.002: (goto_waypoint robot_helper wp5 wp15) [18.000]
33.003: (goto_waypoint robot_assistant wp16 wp36) [32.000]
44.003: (goto_waypoint robot_helper wp15 wp37) [31.000]
65.004: (goto_waypoint robot_assistant wp36 wp38) [8.000]
73.005: (grasp_object robot_assistant post1 wp38) [2.000]
75.004: (goto_waypoint robot_helper wp37 wp48) [9.000]
75.006: (goto_waypoint robot_assistant wp38 wp36) [8.000]
83.007: (goto_waypoint robot_assistant wp36 wp16) [32.000]
84.005: (goto_waypoint robot_helper wp48 wp15) [40.000]
115.008: (goto_waypoint robot_assistant wp16 wp0) [18.000]
124.006: (goto_waypoint robot_helper wp15 wp5) [18.000]
133.009: (place_object robot_assistant post1 wp0) [2.500]
135.510: (goto_waypoint robot_assistant wp0 wp16) [18.000]
142.007: (goto_waypoint robot_helper wp5 wp15) [18.000]
153.511: (goto_waypoint robot_assistant wp16 wp36) [32.000]
160.008: (goto_waypoint robot_helper wp15 wp37) [31.000]
185.512: (goto_waypoint robot_assistant wp36 wp38) [8.000]
191.009: (goto_waypoint robot_helper wp37 wp45) [10.000]
193.513: (grasp_object robot_assistant post2 wp38) [2.000]
195.514: (goto_waypoint robot_assistant wp38 wp36) [8.000]
203.515: (goto_waypoint robot_assistant wp36 wp16) [32.000]
235.516: (goto_waypoint robot_assistant wp16 wp0) [18.000]
253.517: (place_object robot_assistant post2 wp0) [2.500]
```

Introduction and motivation

- Task plans may not be easy to understand by lay users.
- Users are familiar with natural language descriptions.
- We present PlanVerb, a domain-independent method to verbalize task plans for planners based on **PDDL** and **RDDL** domains.
    - Based on verbalization of robot navigation routes by *Rosenthal, Selvaraj, and Veloso (2016)*.

- **Goal: Verbalize both the actions and the causality between them.**

Introduction and motivation

```
0.000: (goto_waypoint robot_assistant wp3 wp0)  [14.000]
0.000: (goto_waypoint robot_helper wp1 wp2)  [7.000]
7.001: (goto_waypoint robot_helper wp2 wp5)  [19.000]
14.001: (scan_place robot_assistant wp0)  [1.000]
15.002: (goto_waypoint robot_assistant wp0 wp16)  [18.000]
26.002: (goto_waypoint robot_helper wp5 wp15)  [18.000]
33.003: (goto_waypoint robot_assistant wp16 wp36)  [32.000]
44.003: (goto_waypoint robot_helper wp15 wp37)  [31.000]
65.004: (goto_waypoint robot_assistant wp36 wp38)  [8.000]
73.005: (grasp_object robot_assistant post1 wp38)  [2.000]
75.004: (goto_waypoint robot_helper wp37 wp48)  [9.000]
75.006: (goto_waypoint robot_assistant wp38 wp36)  [8.000]
83.007: (goto_waypoint robot_assistant wp36 wp16)  [32.000]
84.005: (goto_waypoint robot_helper wp48 wp15)  [40.000]
115.008: (goto_waypoint robot_assistant wp16 wp0)  [18.000]
124.006: (goto_waypoint robot_helper wp15 wp5)  [18.000]
133.009: (place_object robot_assistant post1 wp0)  [2.500]
135.510: (goto_waypoint robot_assistant wp0 wp16)  [18.000]
142.007: (goto_waypoint robot_helper wp5 wp15)  [18.000]
153.511: (goto_waypoint robot_assistant wp16 wp36)  [32.000]
160.008: (goto_waypoint robot_helper wp15 wp37)  [31.000]
185.512: (goto_waypoint robot_assistant wp36 wp38)  [8.000]
191.009: (goto_waypoint robot_helper wp37 wp45)  [10.000]
193.513: (grasp_object robot_assistant post2 wp38)  [2.000]
195.514: (goto_waypoint robot_assistant wp38 wp36)  [8.000]
203.515: (goto_waypoint robot_assistant wp36 wp16)  [32.000]
235.516: (goto_waypoint robot_assistant wp16 wp0)  [18.000]
253.517: (place_object robot_assistant post2 wp0)  [2.500]
```

Building blocks: semantic tags

- PDDL domains are not always self-explanatory (taken from IPC2018 agricola domain):

```
;; This give an extra food is you have an oven
(:action ag__harvest_collecting_fromoven
 :parameters (?r - round ?s - stage ?v - vegetable ?i1 ?i2 ?i3 - num)
...
```

another example:

```
(:durative-action goto_waypoint
 :parameters (?v - robot ?a ?b - waypoint)
...
```

- We need information to generate the sentences relating to every action and their parameters
- We propose to tag each action with the building blocks for the sentence
    - This also may help to document PDDL actions :-)
    - It avoids the need to write full templated sentences, but just the enough information needed to build the sentences afterwards

Building blocks: semantic tags

Example of semantic tags:

- PDDL:

```
; verb = go / travel / move
; subject = ?v
; prep = from the ?a
; prep = to the ?b / towards the ?b !
(:durative-action goto_waypoint
 :parameters (?v - robot ?a ?b - waypoint)
```

- RDDL:

```
// Moves the robot from one waypoint to another
// verb = go / travel / move
// subject = \1
// prep = from the \2
// prep = to the \3 / towards the \3 !
goto_waypoint(robot, waypoint, waypoint): { action-fluent, bool, default = false };
```

This allows to generate **12** different sentences.

Verbalization space

- We extend the notion of verbalization space from *Rosenthal, Selvaraj, and Veloso (2016)*.
    - It defines different variations of the descriptions of plans to cover user preferences.
    - Comprises Abstraction, Locality, Specificity, and **Explanation**.

- Abstraction:
    - 4 levels, from no abstraction (numerical values, real-world coordinates) to full abstraction (only necessary information for sentences to make sense)
- Locality (scope of the narration):
    - All the plan, a range of actions or those actions relating to a subject or object.
- Specificity (level of detail):
    - General picture, summary, detailed narrative

Verbalization space: Abstraction

Level of concretion in the verbalised plans.

- **A1:** No abstraction → Including numerical values (i.e., coordinates), duration and all action parameters.
- **A2:** Parameter names instead of coordinates, durations and all parameters.
- **A3:** Ignores durations, keeps intermediate parameters.
- **A4:** Ignores non-needed parameters and intermediate values.

Verbalization space: Specificity

Level of detail of the verbalised plan.

- **General picture:** Highlights of the plan, only actions achieving goals are verbalised.
- **Summary:** The whole plan, but compressing actions when possible.
- **Detailed narrative:** Detailed description of the whole plan, without compressing actions.

Verbalization space: Locality

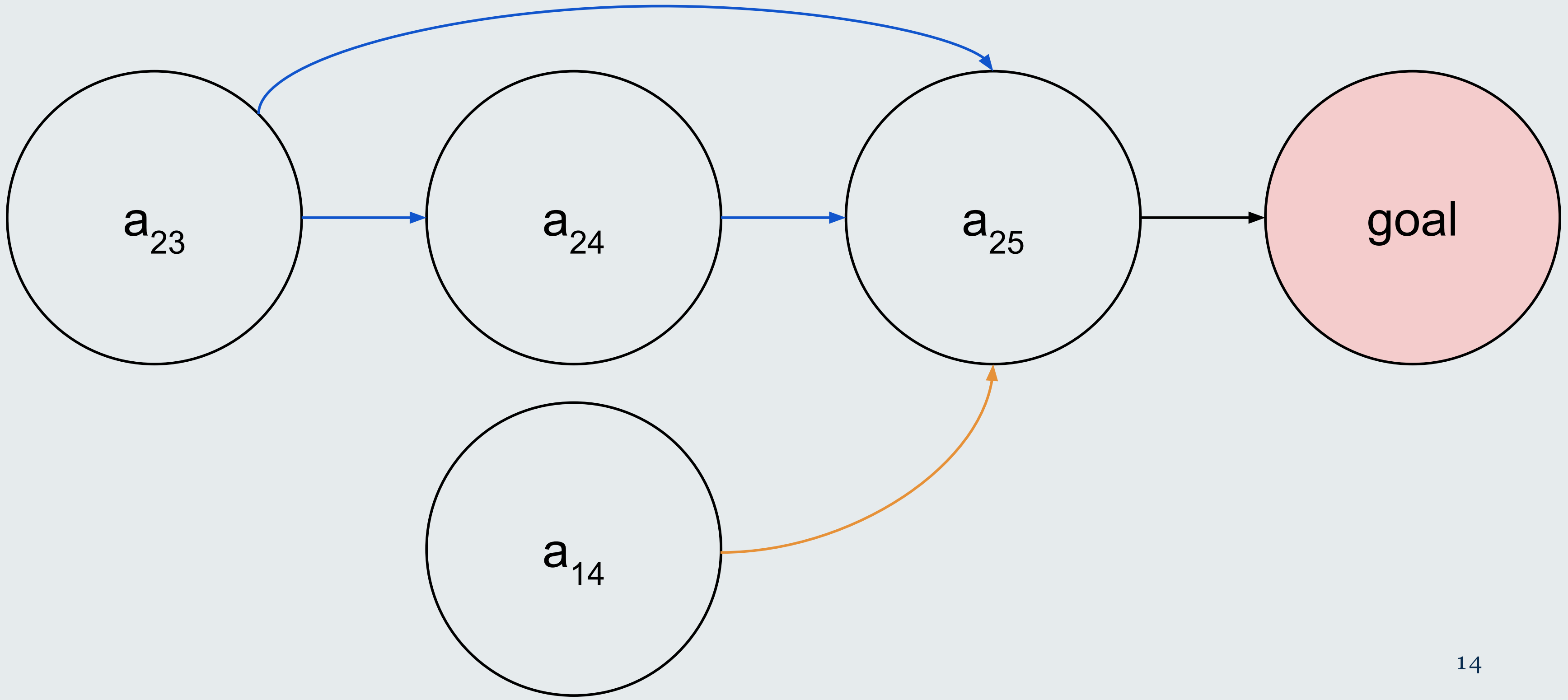
Narrows the verbalisation scope.

- **All plan:** All the actions are verbalised.
- **Range of actions:** Restricts the scope to a subset of actions of the plan.
- **Action or object:** Limits the verbalization to only specific actions or the actions that have a specific object instance as a parameter.

Verbalization space: Explanation (amount of justification)

- We verbalize causal relations and link actions together. We have three kinds of *explanation*:
  - Immediate justifications
  - Deferred justifications
  - Goal-achieving explanations

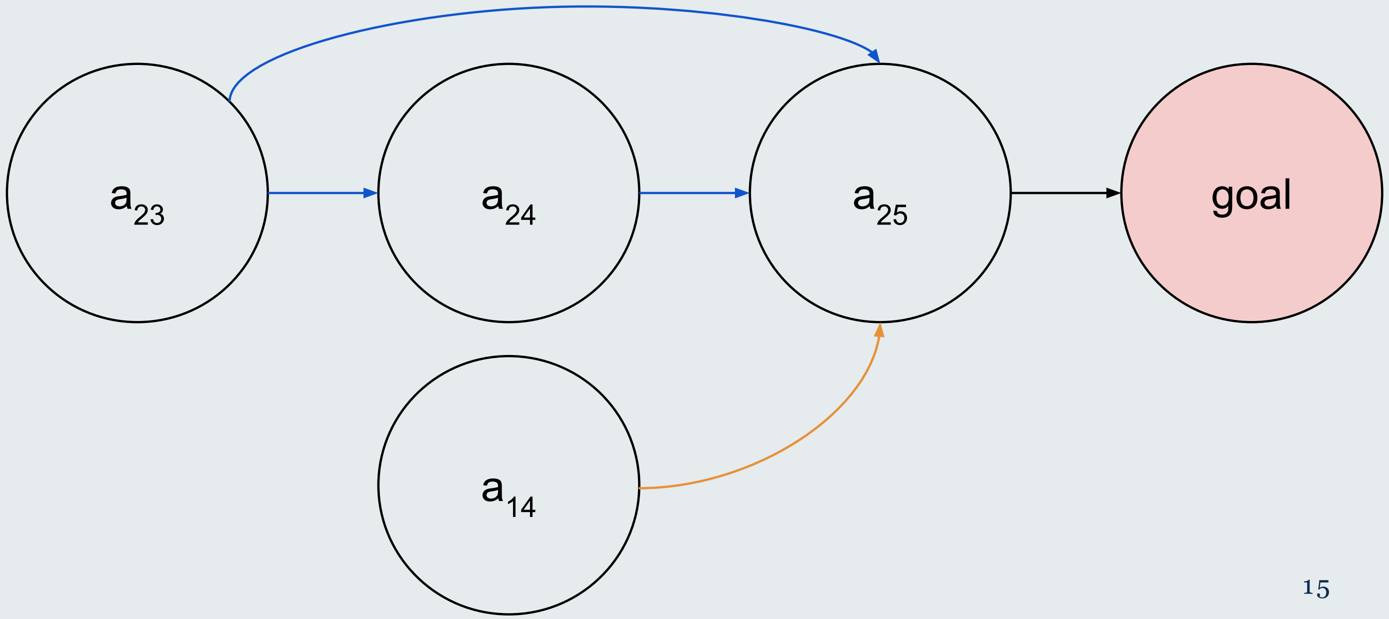
- We define 5 levels:
  - From no explanation
  - to all the justifications



Verbalization space: Explanation (amount of justification)

Explanation parameters:

- **E1:** No explanation.
- **E2:** Verbalizes actions together (making causality explicit) when one action is an immediate justification of another.
- **E3:** Includes deferred justifications when the action achieves a goal.
- **E4:** Goals are verbalized when actions achieve a goal.
- **E5:** All deferred justifications are added.



Plan summarisation through action compression

- We compress actions that may be superfluous or non-informative.

$$\begin{array}{c}\texttt{(grasp r A)}\\ \texttt{(grasp r B)}\\ \texttt{(grasp r C)}\end{array} \rightarrow \begin{array}{c}\texttt{(grasp r [A B])}\\ \texttt{(grasp r C)}\end{array} \rightarrow \texttt{(grasp r [A B C])}$$

$$\begin{array}{c}\texttt{(goto r}_1\texttt{ A B)}\\ \texttt{(goto r}_2\texttt{ A B)}\end{array} \rightarrow \texttt{(goto [r}_1\texttt{ r}_2\texttt{] A B)}$$

$$\begin{array}{c}\texttt{(goto r A B)}\\ \texttt{(goto r B C)}\\ \texttt{(goto r C D)}\end{array} \rightarrow \begin{array}{c}\texttt{(goto r A C)}\\ \text{(through B)}\\ \texttt{(goto r C D)}\end{array} \rightarrow \begin{array}{c}\texttt{(goto r A D)}\\ \text{(through B and C)}\end{array}$$

The PlanVerb algorithm

1. We get a plan, the verbalization space parameters, and the tagged domain as an input, and we then compute causal chains starting from the goals and moving backwards. We use ROSPlan for that.

2. We summarize the plan, joining intermediate actions when needed *(iff verbalization space)*

3. We then compute a causality script, which for each action specifies the actions that enable it immediately and the actions it enables later in the plan, and which goal it achieves (if any).
   - This script is then filtered, and some causal links removed based on the verbalization space parameters.

4. We finally verbalize those actions scripts using semantic tags, making the causality explicit, and conjugating the verbs to the appropriate tense.

All this is performed in a per-subject basis.

Verbalization questioning

- We propose a questioning procedure to get more information on the plan.
- The user introduces a natural language question (i.e., "why did you pick the cup?")
- The grounded PDDL action is matched from the question, disambiguating if necessary, by asking back the user regarding unmatched action parameters.
- The matched action is then verbalised using all the justifications and details (ignoring the any previously set verbalization space parameters).

Examples

Abstraction:

Tomo will locate the manager, which will allow me to **later** request the manager at the kitchen corridor and me to hand post2 to the manager at the kitchen corridor.

Tomo is going to locate the manager, which will allow me to **later** request the manager and me to deliver post2 to the manager.

Specificity:

I will travel from the kitchen shelf towards the kitchen counter (via coffee table 1 and kitchen corridor) so I can leave the paper at the kitchen counter to achieve the goal of the paper being at the kitchen counter.

Rover3 and Rover2 are traveling from waypoint7 towards waypoint0 (taking 5 seconds).

Questioning examples

Q: Why did you grasp the paper?

A: **I took the paper at the desk, which will allow me to later place the paper at the office shelf.**

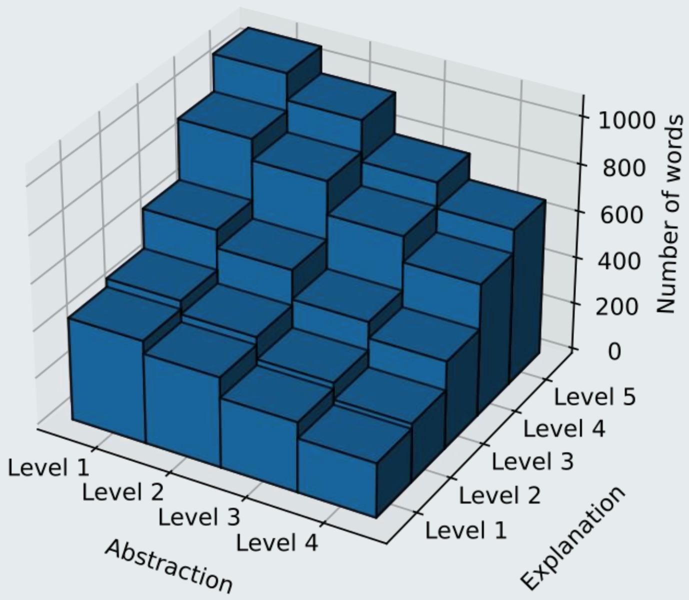
Q: Why will you go to kitchen?

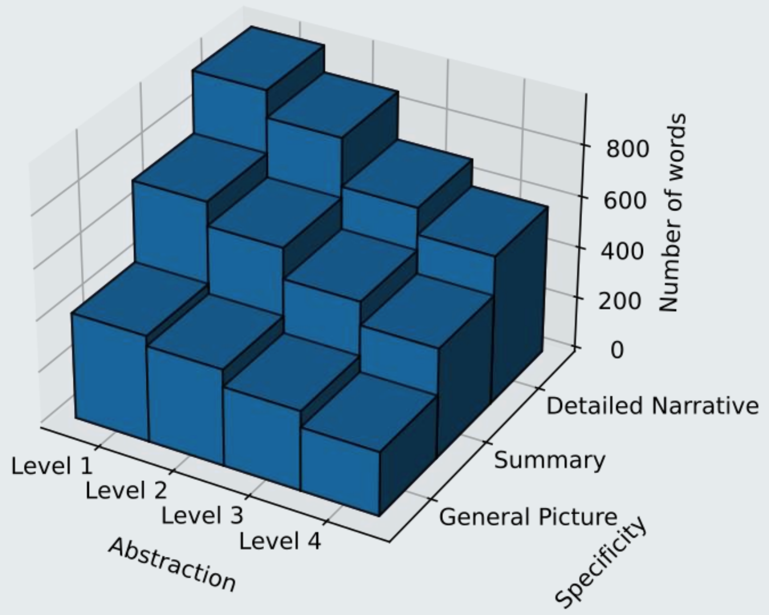
A: Do you refer to when I will go from the entrance or the corridor towards the kitchen?

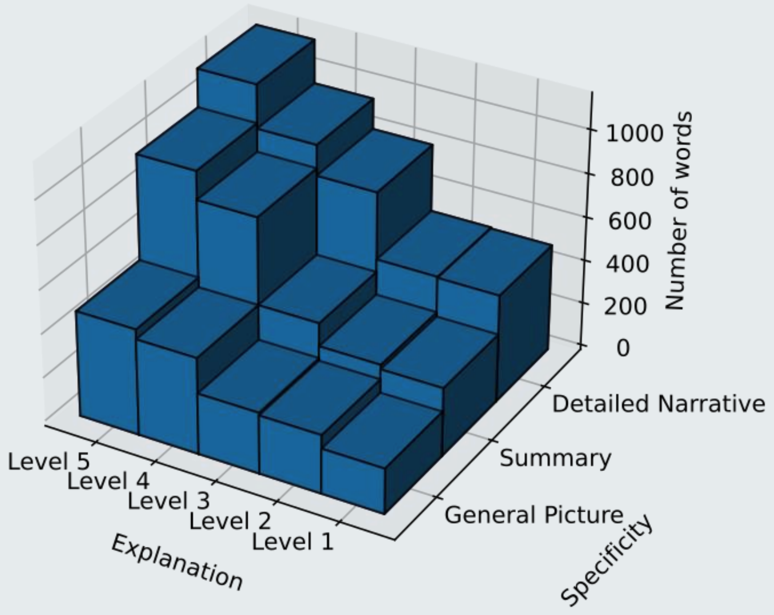
Q: From the entrance.

A: I am going from the entrance towards the kitchen to later be able to check the kitchen, find person1, locate person2, and move from the kitchen to the desk.

Verbalization space parameters







Online user survey

- 42 online users.
- Users saw two verbalizations of the same plan with two robots:
  - v_1: step-by-step plan, without summarization nor explanations
  - v_2: summarized version including explanations

- The users had to answer questions about the plan, and the relations between actions.
- **Results:**
  - Users were able to better answer the questions for v_2 (>80%, with <2.38% answered unknown)
  - Only half of the users were able to answer questions for v_1 (with 20-40% of unknown answers)
  - Confidence on the responses and easiness to answer were higher for v_2
  - Suggestions: verbalization plus visualization

Conclusions

- Making causal information explicit makes it easier for users to understand and reason about a plan.

- Still margin for improvement (future work):
  - Improved text generation. Some sentences are repetitive or result in a strange grammar.
  - Pronominalization could improve the sentences by making them more natural.
  - Integration of preconditions and effects to further explain the reasons behind the actions.

Lennart Wachowiak

Peter Tisnikar

Andrew Coles

Matteo Leonetti

Oya Çeliktutan

KING'S College LONDON

Explanations are nice... but not when they're not needed!

Lennart Wachowiak, Peter Tisnikar, Gerard Canal, Andrew Coles, Matteo Leonetti, and Oya Celiktutan. "Analysing Eye Gaze Patterns During Confusion and Errors in Human–Agent Collaborations". In: 31st IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN). 2022.

A lot can go wrong during collaborations...

Problem

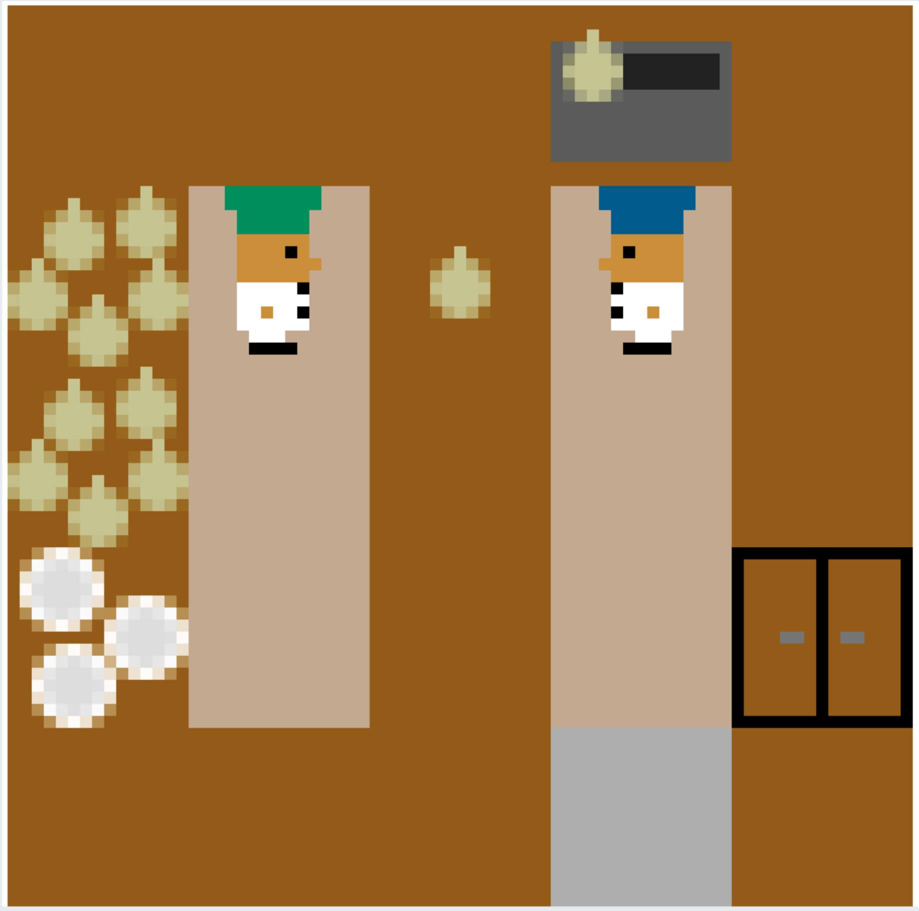
How can the agent know that something is going wrong?

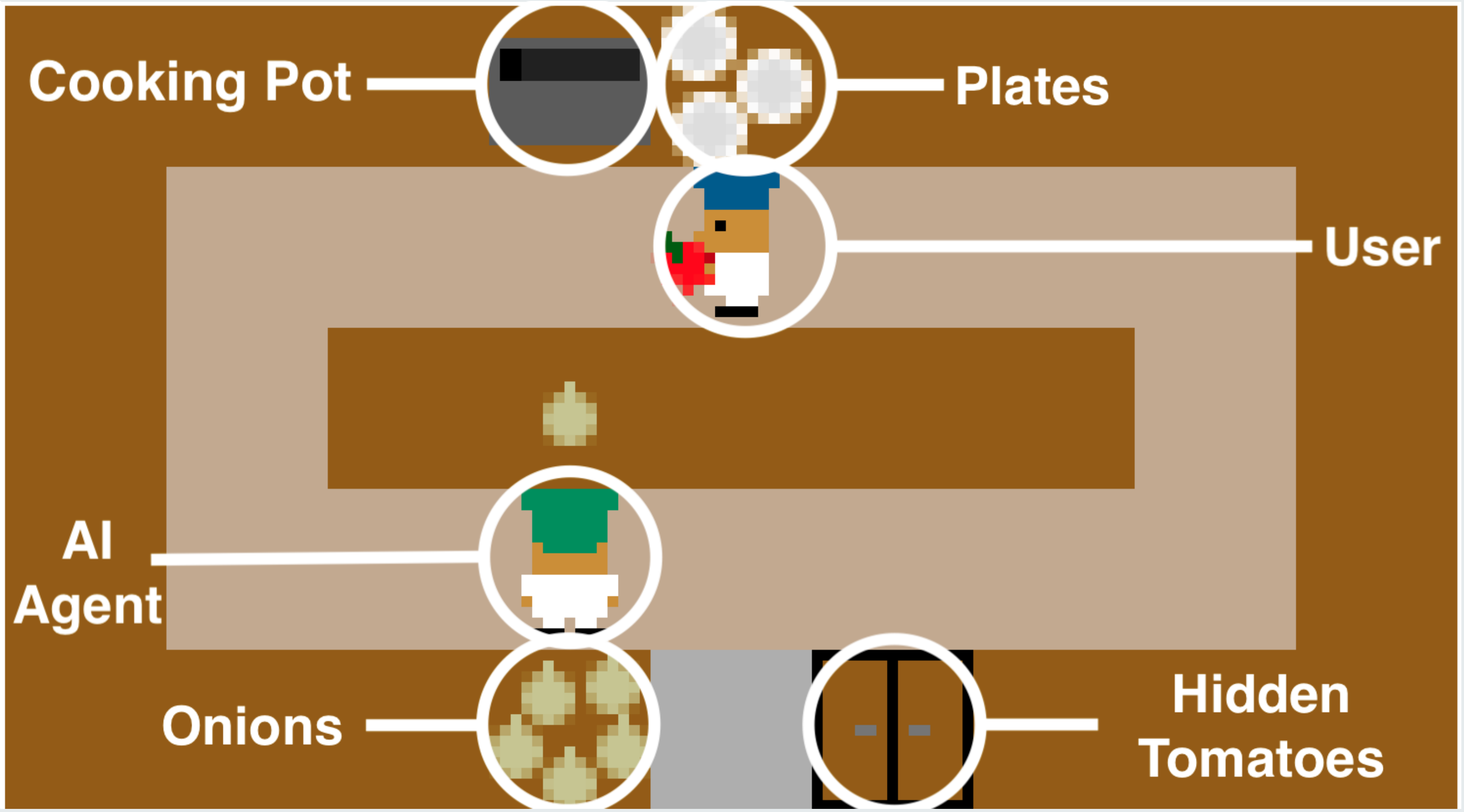
Eye gaze patterns are known to reflect aspects of our mental states!

In this presentation:

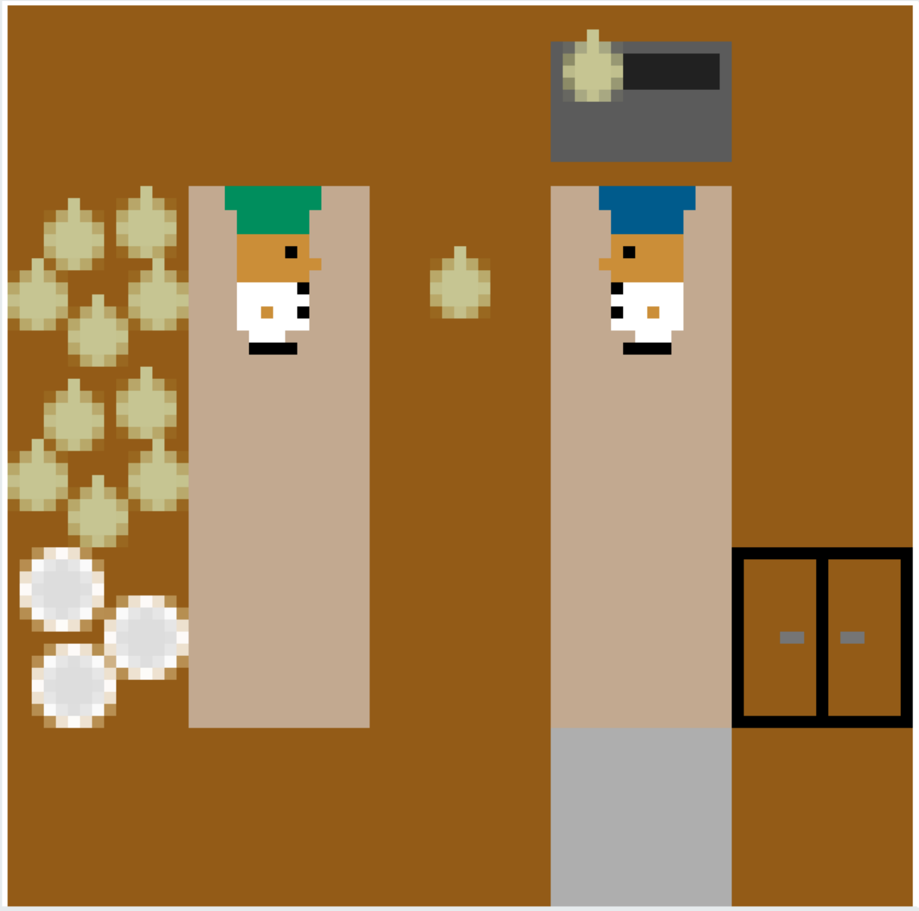
How do **gaze patterns** differ during **user confusion** and **agent errors**?

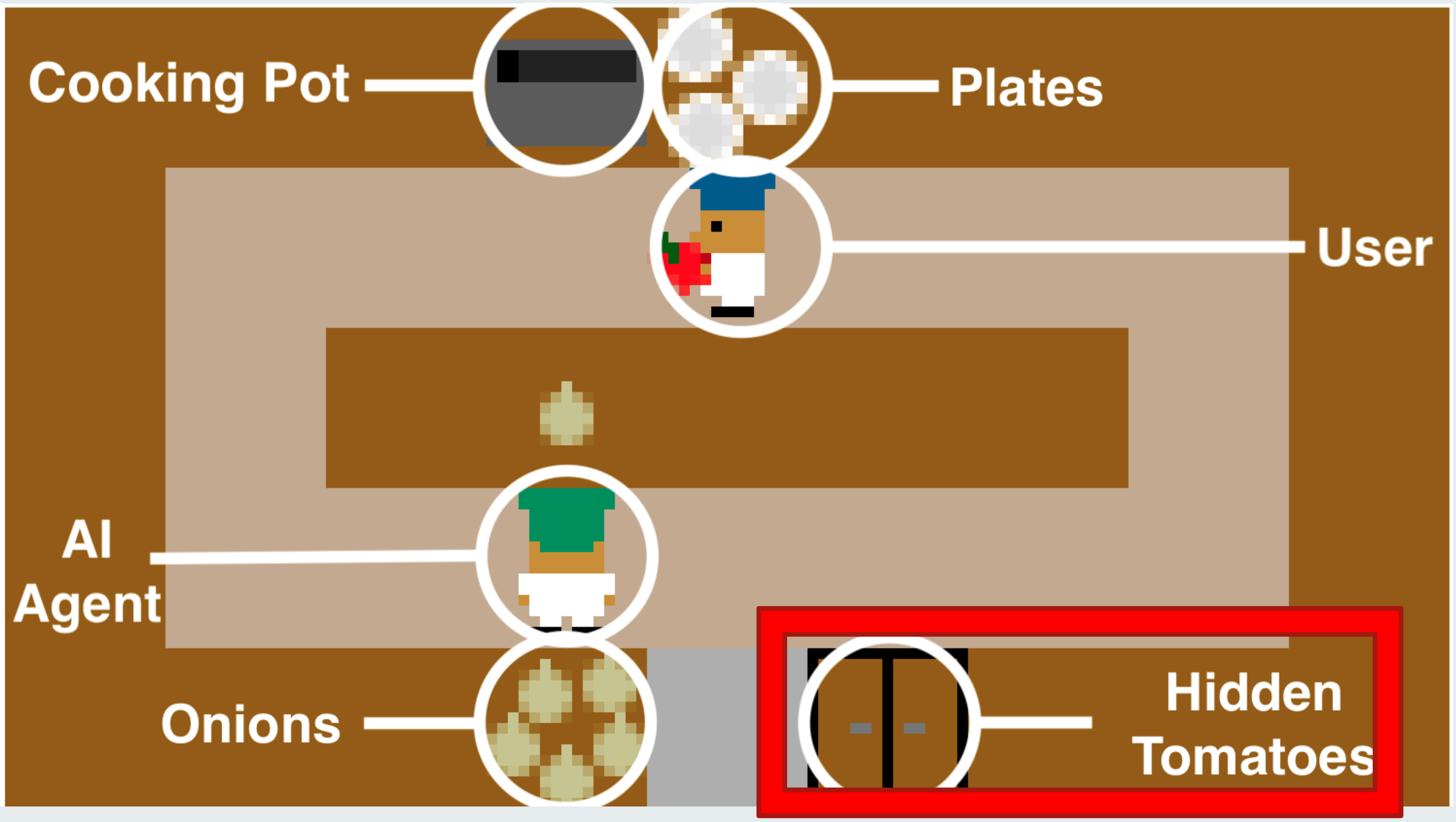
Task



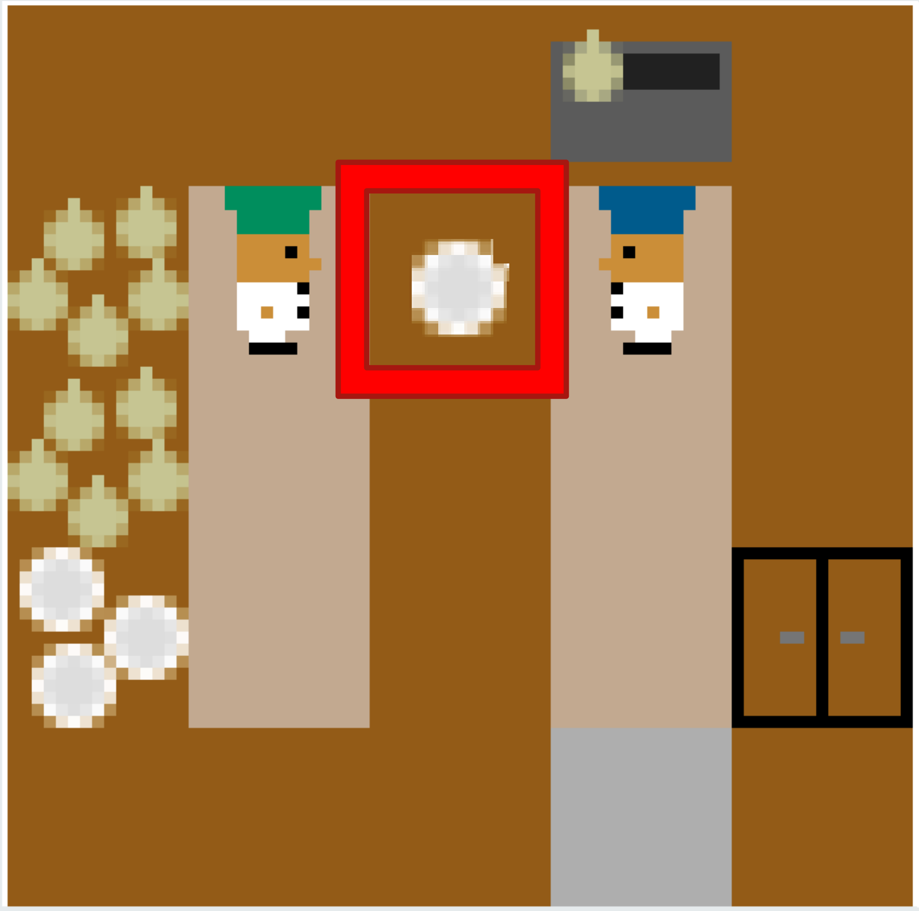


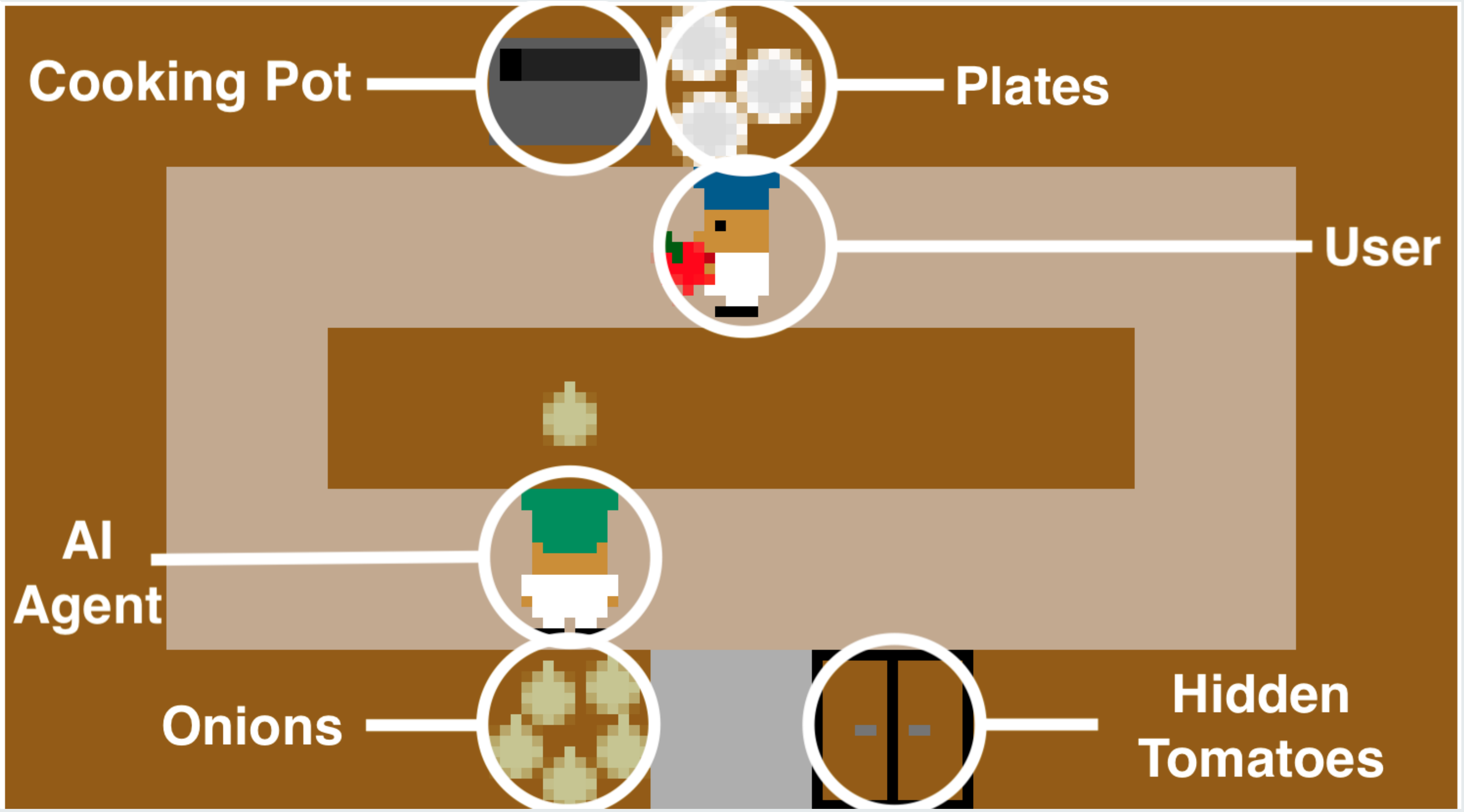
Task – Confusion Triggers

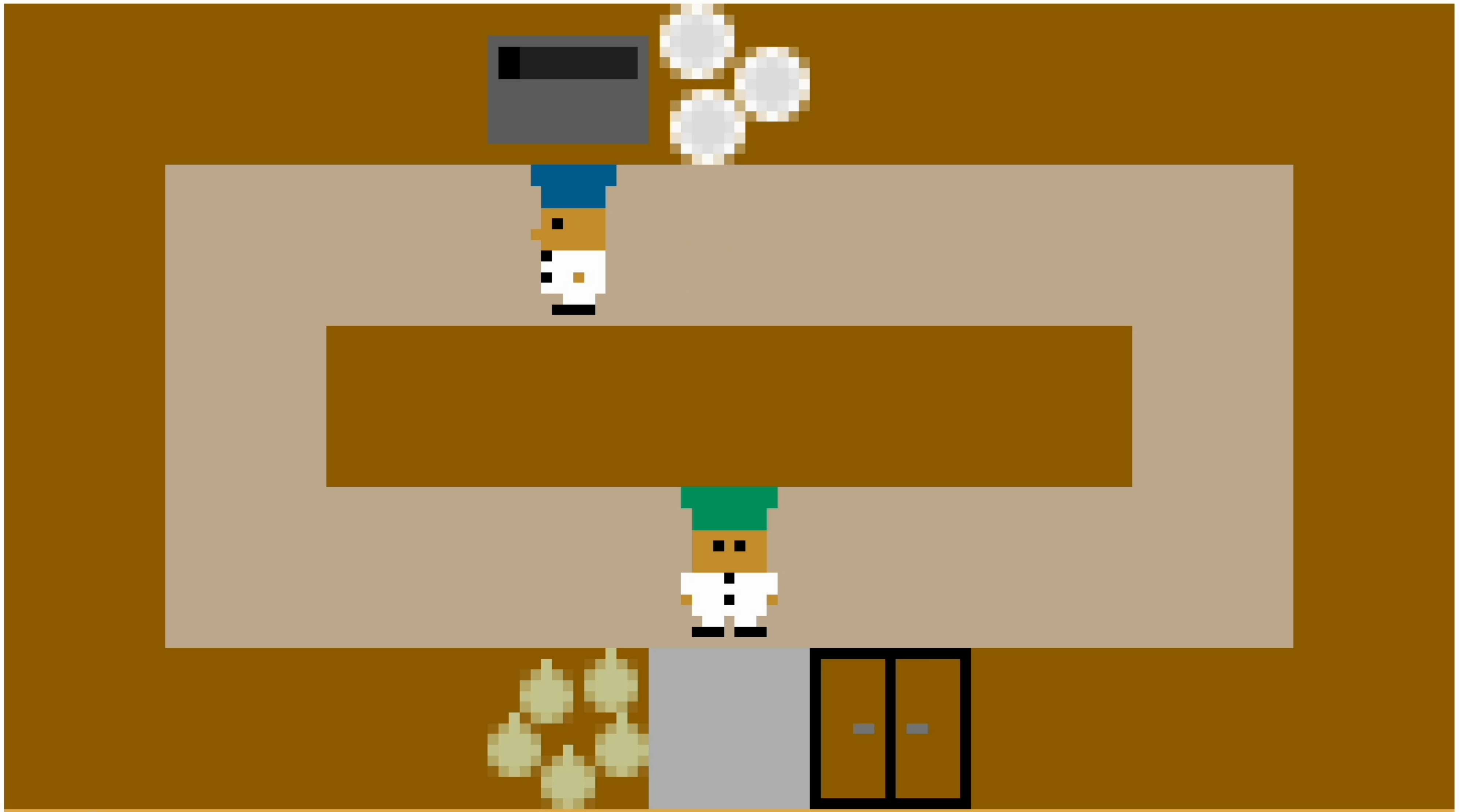




Task – Agent Errors







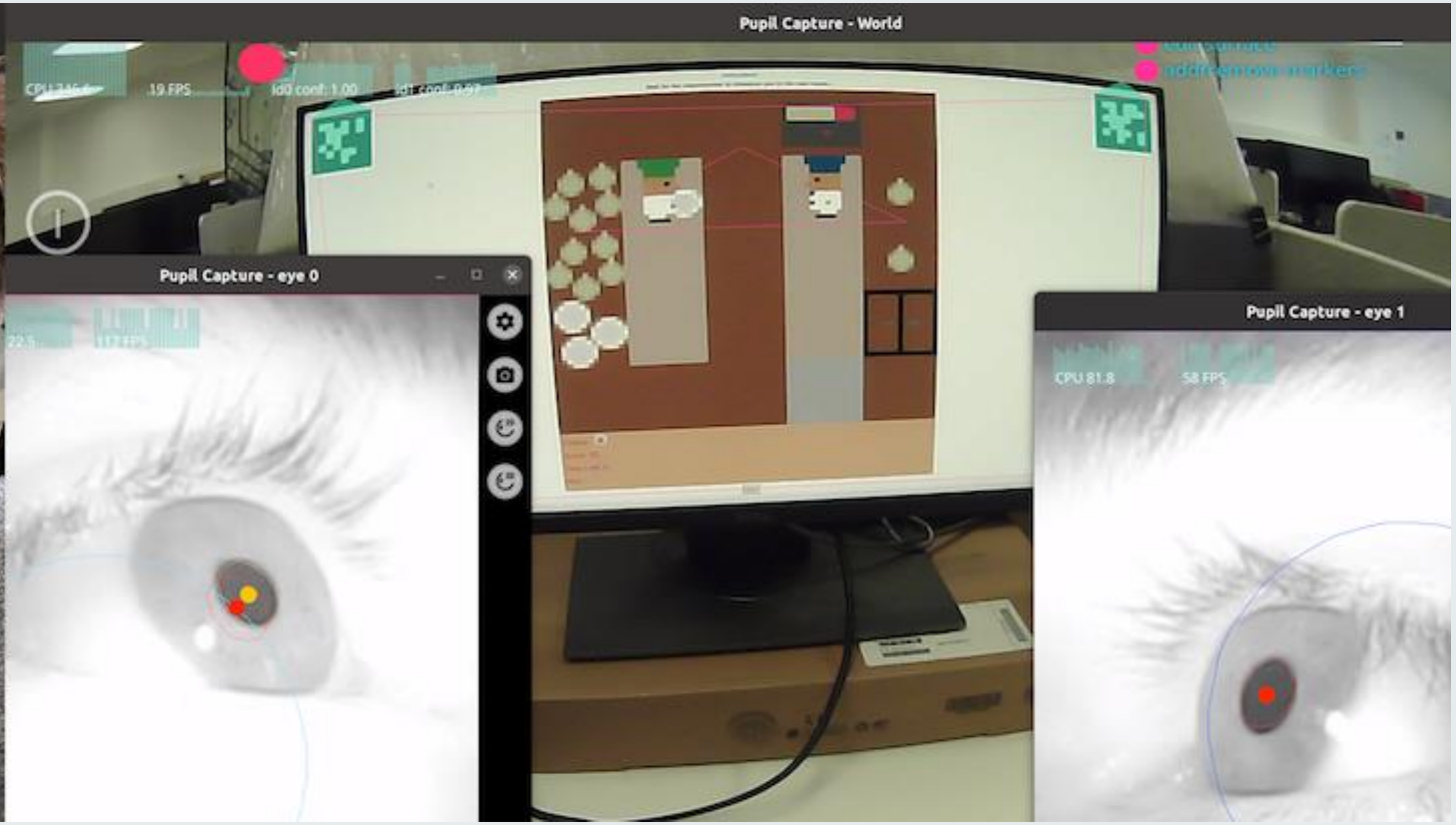
Orders:

Score: 40

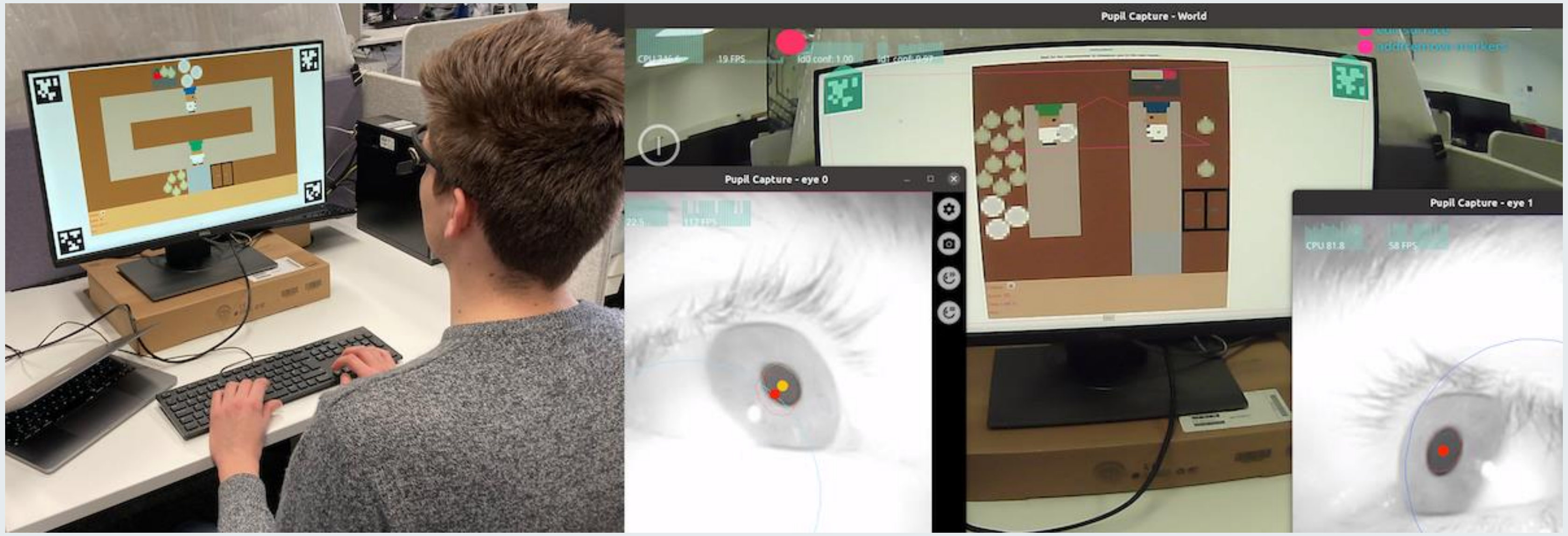
Time Left: 39

Hint:

Setup


Pupil Capture - World
19 FPS
id0 conf: 1.00
Pupil Capture - eye 0
Pupil Capture - eye 1
CPU 81.8
58 FPS

Setup



Data collection with 30 participants

Hypotheses

H1: The user’s gaze will be proportionally more focused on the agent during time-frames in which the agent makes errors compared to those where the agent makes no errors

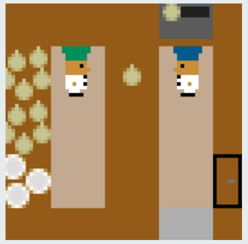
H2: The user’s gaze patterns during time-frames in which they do not know what to do next will differ from when they are sure what to do next

Data Annotation

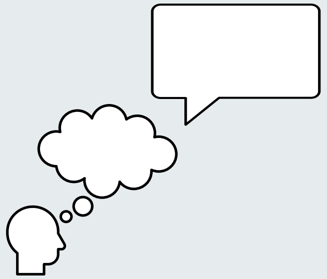
During which frames did the agent make errors?

During which frames were the participants confused?

Data Annotation

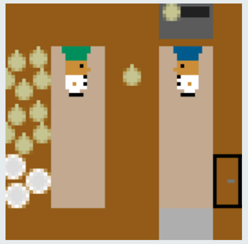


Screen Recordings



Retrospective Think-Aloud Protocols

Data Annotation

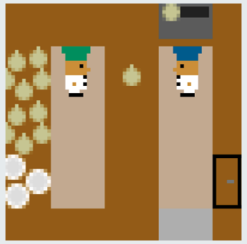


Screen Recordings

Retrospective Think-Aloud
Protocols

Human Annotations for
each frame (Cohen’s Kappa=0.8)

Data Annotation



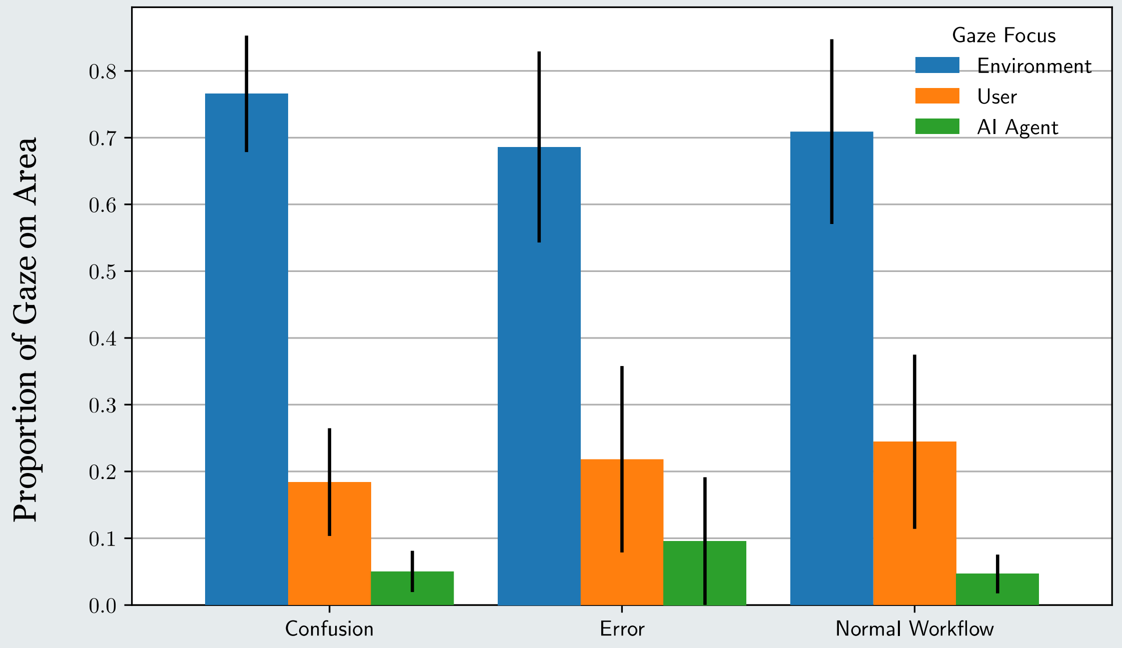
Screen Recordings

Retrospective Think-Aloud Protocols

Human Annotations for each frame (Cohen’s Kappa=0.8)

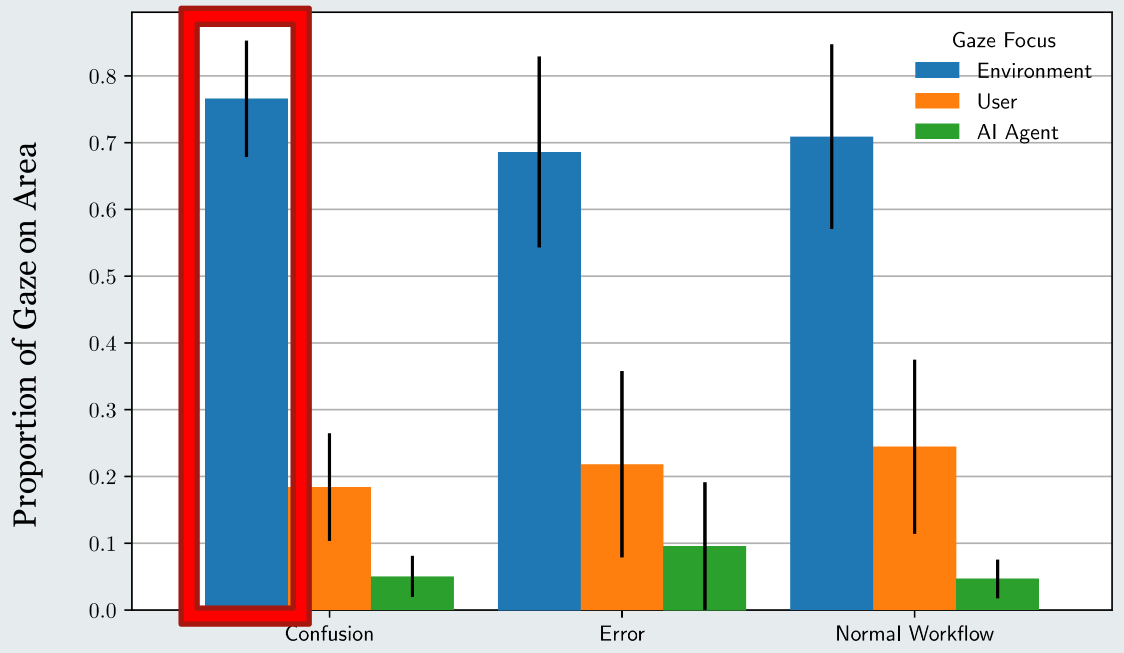
Condition	% of frames
User Confusion	26%
Agent Errors	6%
Normal Workflow	68%

Results


Gaze Focus
Environment
User
AI Agent
Proportion of Gaze on Area
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Confusion
Error
Normal Workflow

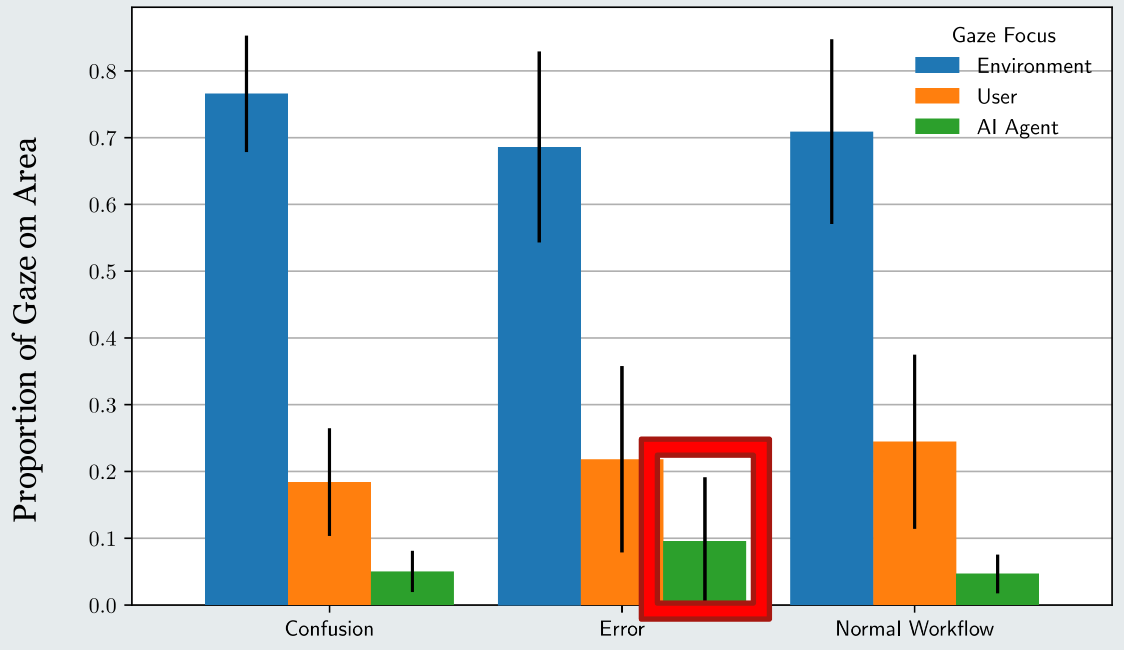
Results

Users look more at the environment when confused



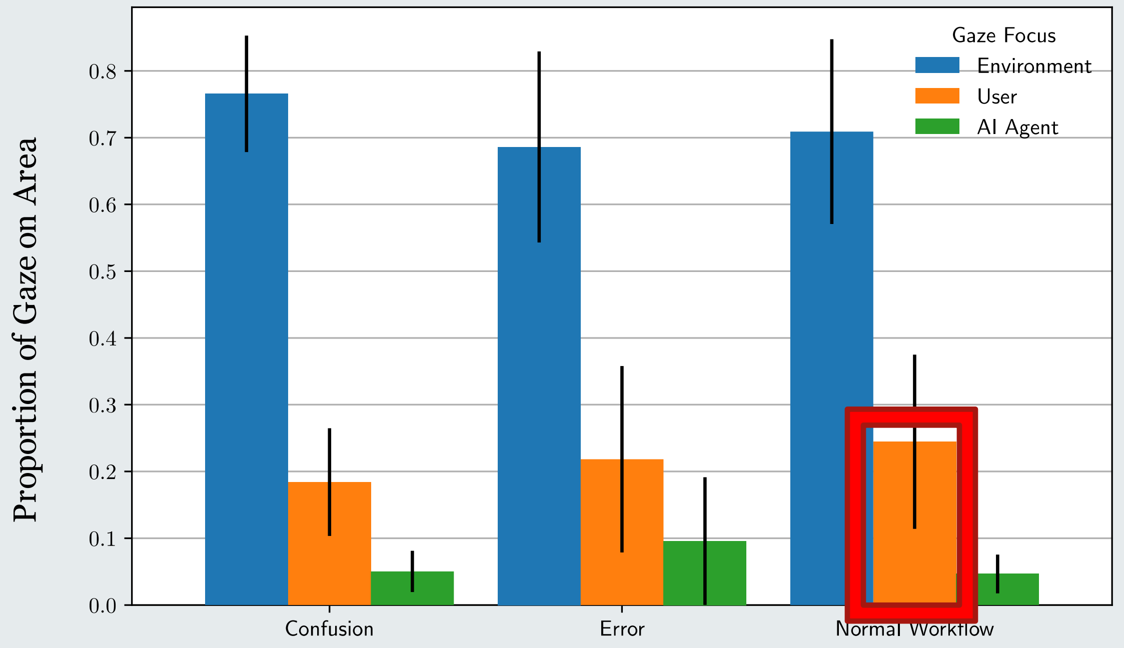
Results

Users look more at the agent during agent errors



Results

Users look more at their own character if everyone runs smoothly



Discussion

Insights from the interviews

- The users not always notice the agent making an error
- More confusion than expected
- Confirm the statistics, e.g., one user during productive workflow said:

  “I didn’t really think about what the agent was doing particularly or worrying about what was happening. I was just thinking about myself”

Future Work

- Extension to physical robot and multiple modalities (**WIP**)
- Evaluation of ML models to predict confusion and agent errors

Conclusions

Gaze patterns can inform us about **difficulties** during an interaction!

- Users look more at the agent when it makes errors
- Users look more at the environment when they do not know what to do

Analyzing such patterns could help an agent to improve its explanations

Take-home message

- Explicit causal information in the explanations may help users better understand plans or sequences of events.

- Determining **when** to explain is almost as important as as to **what** to explain.
  - Proactivity is important!

Thank you for your attention!

I'm glad you asked that, because I wanted to take this time to explain my evil plan.

Gerard Canal

Contact details
Dr Gerard Canal
King's College London
Department of Informatics
Bush House, 30 Aldwych, London, WC2B 4BG
gerard.canal@kcl.ac.uk
https://gerardcanal.github.io