Interactive and Cooperative Delivery of Referring Expressions: A Comparison of Three Algorithms

Jana Götze **Karla Friedrichs** **David Schlangen**
Computational Linguistics, Department Linguistics, University of Potsdam, Germany
{jana.goetze,friedrichs,david.schlangen}@uni-potsdam.de

Abstract

In interaction, the establishment of reference is a collaborative process involving the main speaker and the addressee. Current work on visual natural language generation however minimizes interactivity and concentrates on the complexity of the input. Here, we return to some classical rule-based NLG algorithms, and extend them minimally to achieve incremental referring behavior guided by the listener's non-verbal feedback in a visual domain. We run a human evaluation study and show that these algorithms create behavior that is effective, though not judged as human-like. An additional, even simpler algorithm that generates finer-grained instructions is shown to be even more effective in ambiguous settings. We speculate that such simple algorithms can act as teachers that can help neural models take a step towards interactivity.

1 Introduction

In interactive settings, the establishment of reference to objects is a collaborative process, shaped by the referrer as well as the addressee. Even though this is by no means a new insight (e.g., Clark and Wilkes-Gibbs (1986); Heeman and Hirst (1995)), it is one that has moved outside the focus of much current work on visual natural language generation, which concentrates on the complexity of the input (e.g., raw image data instead of symbolic representations of the visual context) and minimizes interactivity, even if the chosen name of the task, e.g., Das et al. (2017)'s "visual dialog" or Savva et al. (2019)'s "embodied AI", might suggest otherwise (Benotti and Blackburn, 2021).

In task-oriented dialog, subdialogs emerge when an instruction follower (IF) asks for clarification in case they are unsure. Even when the IF does not interact verbally, the instruction giver (IG) collaboratively guides the IF after giving an initial instruction by iteratively providing feedback and additional information (Striegnitz et al., 2012). Figure 1 shows an example from a human-human data collection.

Figure 1: Example for a task-oriented interaction in shared visual space; cf. (Zarrieß and Schlangen, 2018).

Here, we investigate whether such non-verbal user behavior can be used in combination with classical rule-based Referring Expression Generation (REG) algorithms — the Incremental Algorithm of Dale and Reiter (1995); and Denis (2010)' Reference Domain Theory — for continuously providing feedback to the IF in an object identification task. We evaluate the resulting interactive algorithms in human evaluations, and show that they create behavior that leads to high task success. Humans evaluate none of the algorithms as human-like but accept them as reasonably likeable, friendly and competent. An additional, even simpler algorithm that generates finer-grained instructions is shown to be even more effective in ambiguous settings and is slightly favored overall by participants.

We close with speculations on how such rule-based systems that use symbolic input could be used as data generators for more flexible learning-based systems that combine robustness on the input side with more natural grounding behavior.

2 Related Work

Incorporating non-verbal listener feedback into REG systems has been the subject of previous studies. Especially eye gaze has been interesting to investigate in this context as studies in psycholinguistics have shown that listeners attend to objects

Proceedings of the 26th Workshop on the Semantics and Pragmatics of Dialogue, August, 22-24, 2022, Dublin.

in their visual environment as they are being referred to (Tanenhaus et al., 1995).

In task-oriented interaction data in the context of the GIVE challenge (Byron et al., 2007), which provided a 3D environment in which instruction followers (IF) moved around, eye gaze has been found to be a good predictor of what object the listener resolved a referring expression (RE) to (Engonopoulos et al., 2013; Koleva et al., 2015; Staudte et al., 2012). Koller et al. (2012) have integrated eye gaze and movement information directly into their REG algorithm to produce positive and negative feedback and found that eye gaze improves referential success most but that also movement information was useful compared to giving no feedback at all. Both feedback systems reach high task success rates but required interaction data from humans for training. In our work, we want to investigate the suitability of existing rule-based algorithms that require no previous training data. Instead of eye gaze data, we rely on positional information of the listeners' movement which has also improved task success in Koller et al. (2012)'s experiments.

We will investigate two rule-based algorithms: the Incremental Algorithm (IA) (Dale and Reiter, 1995) and an algorithm based on Reference Domain Theory (RDT) (Denis, 2010). While the IA assumes the full context of objects to be available when generating a RE, RDT includes a notion of focus on subsets of objects and makes it suitable for environments in which the listener's view changes. We carry out our experiments in a 2D environment where the listener has access to the full set of objects and the IG – the REG algorithm – can "see" what the IF is doing.

Models that account for the explicit collaborativeness of reference have been proposed as well. For example, Heeman and Hirst (1995) use a planning-based approach that accounts for clarification requests as modifications of the plan and allows each partner to modify the plan – the referring expression – directly. We leave this extension to future work.

More recently, researchers have attempted to generate instructions and descriptions based on images, bypassing the need to create a symbolic representation of the domain, and thus being able to leverage the capabilities of modern neural network models (Das et al., 2017; Savva et al., 2019). However, these efforts do not account for the collaborative nature of reference even if the task names may suggest otherwise (Benotti and Blackburn, 2021). They instead separate the generation and evaluation of reference from one another without allowing for a collaborative modification of the generated RE. Instructions however need to go beyond correctness in that the description attempts to elicit the desired behavior in the listener. In order to obtain suitable data for training a neural network model, we therefore need to make sure that the input language data is both correct and suitable for the task. We investigate whether rule-based algorithms are a possible data generation mechanism by testing their generated output in human evaluation in a domain that we can access in both symbolic and continuous format.

3 Rule-based collaborative instructions

For this research, we adapt and extend the Incremental Algorithm (IA) (Dale and Reiter, 1995) and the Reference Domain Theory (RDT) (Denis, 2010), and set up an additional algorithm called Supervised Exploration (SE), which we explain in this section. All algorithms generate an initial referring expression based on the current visual context and then continuously monitor user behavior to provide continuous feedback to the IF. All three implementations are available at `https://github.com/kfriedrichs/golm/tree/ba`.

In order to select an object from the set, the instruction follower moves towards the object via a *gripper* – a cursor that can be controlled using the keyboard. This movement constitutes the user behavior that each algorithm bases its feedback on. After the initial RE, each system monitors gripper movement and generates either positive or negative feedback (YNFEEDBACK), or, when no movement has happened for a certain time, an adjusted RE (a new GENERATERE event) depending on the specific algorithm.

Each system monitors the gripper movement as well as the time to detect idle times. When the gripper has moved three grid units, YNFEEDBACK is generated based on the movement with respect to the target object. When the gripper has moved less than three units in 10 seconds or the participant has gripped an incorrect object, a new GENERATERE instruction is produced.

Algorithm 1 shows pseudocode for the general procedure that was used to instantiate each specific algorithm and the following feedback mechanism.

Algorithm 1 Event-driven feedback mechanism.
In the experiments, we set: Timeout=10sec, Threshold=3 units on grid, MaxTries=3

```
1:  procedure ON NEWTASKEVENT
2:      GENERATERE                                  // IA, RDT or SE
3:  end procedure
4:  procedure ON GRIPPERUPDATEEVENT
5:      if TargetGripped or MaxTriesReached then    // Skip to the next configuration when the
6:          NEWTASKEVENT                            ... correct object was picked or after 3 tries.
7:      else if IncorrectSelection then
8:          THATWASINCORRECT
9:          GENERATERE                              // repeat/rephrase
10:     else if MovedPastThreshold then             // When the gripper has moved in one direction
11:         YNFEEDBACK                              ... for a certain distance, give feedback.
12:     end if
13: end procedure
14: procedure ON TIMEOUTEVENT                       // If nothing has happened for too long
15:     if Gripper moved since Timeout/2 then       // If the gripper has moved recently
16:         YNFEEDBACK                              ... give feedback
17:     else
18:         GENERATERE                              // repeat/rephrase
19:     end if
20: end procedure
```

3.1 Incremental Algorithm

We implement the Incremental Algorithm (IA) as described in (Dale and Reiter, 1995) and extend it by the feedback loop as described above. The IA assumes a preference order of available properties that is known to influence the performance of the algorithm (van Deemter et al., 2012). We set the order to *color–shape–location* based on human RE from existing corpora in the same domain (Zarrieß et al., 2016). The IA will repeat its initial instruction in the case of a new GENERATERE decision.

The algorithm works as follows. It starts with all entities except the target as the *contrast set* and iterates through the given preference order of attributes. Each property that the current target has and that rules out some competing entity is immediately added to the RE, reflecting the *greedy* strategy. Ruled out entities are removed from the contrast set. The expression is complete and returned as soon as all distractors have been eliminated and the set is empty.

The YNFEEDBACK function is implemented here as a random selection from a fixed set; negative feedback is one of ["Not this direction", "Not there", "No"], positive feedback is one of ["Yes, this direction", "Yes", "Yeah", "Yes, this way"].

3.2 Reference Domain Theory

We implement a version of the algorithm based on Reference Domain Theory (RDT) as described in (Denis, 2010). RDT dynamically creates *reference domains* from the available object properties and accounts for discourse *salience* once a RE has been introduced as well as listener *focus* once the IF starts moving. This allows the algorithm to produce underspecified expressions like one-anaphora. We use the gripper movement to account for the listener's focus. The order of properties is set to be the same as for the Incremental Algorithm. Note that RDT uses an additional notion of location in its feedback generation: aside from the regular *location* property describing an object's global position, feedback may specify the position relative to the IF's gripper. Table 1 shows an example for how focus is used to dynamically generate underspecified RE with the RDT algorithm.

3.3 Supervised Exploration

The *Supervised Exploration Algorithm* (SE) generates instructions that are underspecified. It only verbalizes location information and then relies on guiding the IF using continuous feedback without verbalizing any further properties. Since this algorithm never produces a full RE, it continuously

Algorithm 2 Pseudocode of the feedback loop of the Supervised Exploration algorithm. The general feedback behavior in Algorithm 1 is extended by an additional check for whether the gripper is close to the target.

```
1:  if InXRange(TARGET) and InYRange(TARGET) then  // If the gripper is close to the target
2:      return TAKE(TARGET)                       ... issue an instruction
3:  else if MovingInRightDirection then           // If the gripper is moving
4:      return POSFEEDBACK                        ... issue positive and negative
5:  else if MovingInWrongDirection then           ... feedback based on its direction
6:      return NEGFEEDBACK
7:  else if IDLE and InXRange(TARGET) then        // If the gripper is still but close to
8:      return MOVE(y)                            ... the target on one axis, issue
9:  else if IDLE and InYRange(TARGET) then        ... an instruction for the other axis
10:     return MOVE(x)
11: end if
```

Task 2:	*There are multiple blue "W"s on the board. The leftmost is the target piece.*
Agent:	Select a blue W in the bottom.
User:	*moves the gripper towards the incorrect objects on the right*
Agent:	**Not these ones**. Get the blue W in the bottom of the board and left of the gripper.

Table 1: Possible interaction between the RDT agent and a user. The user's gripper movement is used to model their focus, enabling the algorithm to generate the **bold-faced underspecified instruction**. At the time of the second message, only non-target objects matching the initial ambiguous instruction are in the user's focus, therefore *"these ones"* suffices as a description of the negated objects.

checks whether the gripper has already reached its correct position along one of the axes to generate an additional STOP message.

The method is motivated by observations of human-human interactions that achieve object identification without the use of full REs. In some instances, the IF took a trial-and-error approach, continuously trying to guess the next referent or action and consequently receiving feedback from the IG. With this new algorithm, we explore a feedback-only reference strategy for an artificial instruction giver.

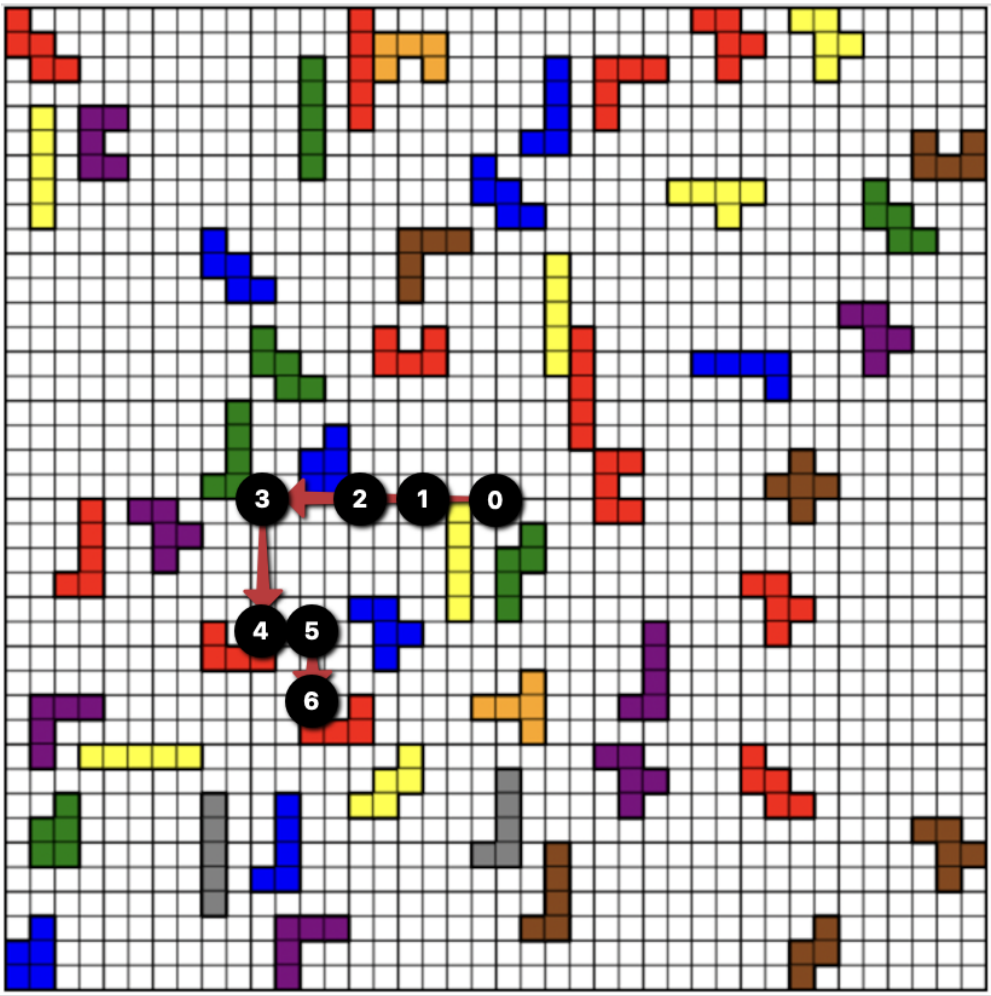
SE solely uses *location* as an attribute. Initially, it generates an instruction to move in one direction, starting with the x-axis. During the feedback loop, moving towards the target is supported by positive feedback, moving away is encountered by negative feedback, using the same fixed phrases as IA. Once one coordinate is in the target's range, *"Stop"* is output, followed either by a direction for the remaining axis or by the instruction to select the object. Since *stopping* the gripper is time-sensitive in order to not move past the object, SE uses an additional feedback trigger activated by the gripper entering the target range on one axis. Pseudocode for the algorithm's feedback loop is shown in Algorithm 2.

3.4 Example and comparison of the algorithms

Our domain is an online puzzle game in which Pentomino shapes have to be selected from a board of many pieces. Figure 2 shows an episode in which the three algorithms vary in their initial instructions.

The difference between IA and RDT is subtle and arises because the available attributes (shape, color and location) do not suffice for a discriminating description – two Pentomino pieces match the phrase "*red U in the bottom left*". Following (Dale and Reiter, 1995), this situation causes a failure of IA. In our experiment, we still used the final expression, which includes all features, for an instruction. Since IA assumes a discriminating RE, a definite article is used. RDT on the other hand acknowledges that the description is ambiguous by inserting the indefinite article to create an explicitly underspecified RE.

The feedback behavior of each algorithm is showcased in the same Figure 2. RDT clearly provides the most detailed feedback: at step 1, all algorithms reinforce the IF's moving direction, but RDT also provides a RE with the additional *location-relative-to-gripper* attribute. Still at 1, a



	IA	RDT	SE
0	— — — — — — *Initial state* — — — — — —		
	Take the red U in the bottom left.	Take a red U in the bottom left.	Go a bit left.
1	Yes.	Yeah. A red U in the bottom left of the board and below left of the gripper.	Yes.
	— — — — — — *No movement for 10 seconds.* — — — — — —		
	Take the red U in the bottom left.	Look for a red U in the bottom left.	Go a bit left.
2			Stop. Go a bit down.
3	Not there.	Not this one. Look for another one.	No. Go a bit right.
	— — — — — — *The user selects an incorrect object.* — — — — — —		
4	That was incorrect. Take the red U in the bottom left.	That was incorrect. Look for a red U in the bottom left.	That was incorrect. Go a bit right.
5			Stop. Go a bit down.
6			Take this object.
	— — — — — — *The user selects the correct target object.* — — — — — —		

Figure 2: Example episode including initial instructions (0) and the feedback (starting at 1) given by each algorithm. The gripper was moved along the arrows. At *1*, the gripper was halted until the feedback timeout triggered. At *4*, an incorrect object was selected. The SE algorithm continuously monitors whether the gripper gets close to the target to issue a STOP message.

new GENERATERE action features a full instruction for all algorithms by definition. Step 3 demonstrates RDT's strength: using one-anaphora, the algorithm acknowledges the presence of identical objects and tries to disambiguate them based on the IF's focus. At step 4, all algorithms output "That was incorrect" followed by a GENERATERE action as before.

The example also shows the increased feedback frequency of SE. At steps 2, 5, and 6, the gripper gets close to the target on at least one axis, triggering an instant *"Stop"* or *"Take this object"* response of the agent.

Note that there might be more feedback messages from each algorithm depending on the movement speed of the gripper, typically at least another YNFEEDBACK between 4 and 6.[1]

4 Experiments

4.1 Method and Procedure

We designed an interactive object identification task in which participants see a playing board online in their browser. The board contains 50 Pentomino puzzle pieces on a 40x40 grid. The pieces can have one of 12 different shapes and 8 different colors and can be rotated and mirrored. Figure 2 shows an example. We design 12 different episodes, one for each of the 12 different Pentomino shapes as target piece. Each participant sees all 12 episodes in the same order.

In order to select pieces, participants use the arrow keys on their keyboard to move a *gripper* depicted by a cross. The gripper is initially positioned in the center of the board for each new episode and can be moved in steps of 0.5 units of the visible grid. In order to select a piece when the gripper touches it, participants use their *space* or *enter* key.

We create 6 *hard* and 6 *easy* configurations. In the easy configurations, the target piece has at least one unique property. In the hard configurations, more than one piece will match the initial instruction, even when all attributes are specified. We achieve this by placing copies (or rotated copies, as rotation is not used as an attribute here) of the target next to the target piece (cf. Figure 2 for an example hard episode). We generate instructions in English, using each of the three algorithms described in Section 3. The instructions are generated offline for each configuration and synthesized using the Amazon Polly TTS standard Matthew voice.[2]

Each participant was randomly assigned one of the algorithms. The data collection starts with an audio test in which the participant is asked to transcribe a phrase that they hear in order to ensure that they could play audio in their browser. Participants are then presented a trial episode in which they could familiarize themselves with the interface.[3]

We log each gripper movement and instruction event in a json format. Timestamped logs are sent to our self-hosted server at the end of the interaction. We use the logged information to derive the following metrics:

Number of incorrect attempts for each episode. The maximum number of trials in each episode is 3, which the participants were informed about during the training episode. After the third trial, the participant sees the next configuration regardless of success. A value of 2 or fewer incorrect attempts reflects task success.

Time to solve an episode in seconds, starting at the end of the initial spoken instruction until the correct grip or third grip.

Number of feedback messages for each episode, i.e. how many times the algorithm verbally reacted to a participant's behavior.

Subjective ratings using 7-point Likert scales in a post-task questionnaire to measure participants' perception of the agent. Throughout the data collection, the voice was referred to as "Matthew" in order to give the agent an identity.

4.2 Results and Discussion

We collect a convenience sample as part of a student project, recruiting primarily university students via email. Participants were unaware of the specific research question. They did not receive reimbursement, but participated in order to support the project. Participation was anonymous.

91 subjects participated. Data from 1 participant was removed because they did not pass the audio test. Of the remaining 90 participants, 43 were female, 41 male, 2 non-binary and 4 did not report,

[1] Apart from the user's speed in moving the gripper, the movement speed also depends on the fire rate of keyboard events. Since the setting did not allow us to control the system setup of each participant, we acknowledge there might have been some variance.

[2] https://aws.amazon.com/polly/

[3] The data collection interface is available at https://github.com/clp-research/golmi.

Algorithm	Success Rate			# Failed attempts			# Feedbacks			Task length			# Episodes		
	all	easy	hard	all	easy	hard	all	easy	hard	all	easy	hard	all	easy	hard
IA	0.94	**0.96**	0.92	0.52	**0.19**	0.85	3.46	**2.56**	4.35	**11.59**	**10.07**	**13.10**	331	164	167
RDT	0.93	0.95	0.90	0.58	0.27	0.89	**3.07**	2.73	**3.41**	12.91	11.25	14.57	351	174	177
SE	**0.95**	0.95	**0.95**	**0.29***	0.33	**0.24***	5.93*	6.18*	5.68*	14.98*	15.58*	14.37	384	192	192

Table 2: Summary of results by episode type. The task ends when the correct piece is gripped or after 3 attempts. The task is successful if there were 2 or fewer failed attempts. Task length is reported in seconds. *indicates a statistically significant difference between the result for SE and both IA and RDT (ind.t-test, p<0.001).

Dimension	IA	RDT	SE
machine-like – human-like	2.43	2.67	**2.75**
incompetent – competent	4.29	3.40	**4.81**
dislike – like	4.25	3.73	**4.28**
unfriendly – friendly	4.46	4.40	**4.97**
unpleasant – pleasant	**4.29**	3.63	4.25

Table 3: Results from the post-task questionnaire. All scales ranged from 1 to 7 in the order of the specified adjectives. $N_{IA} = 28$, $N_{RDT} = 30$, $N_{SE} = 32$.

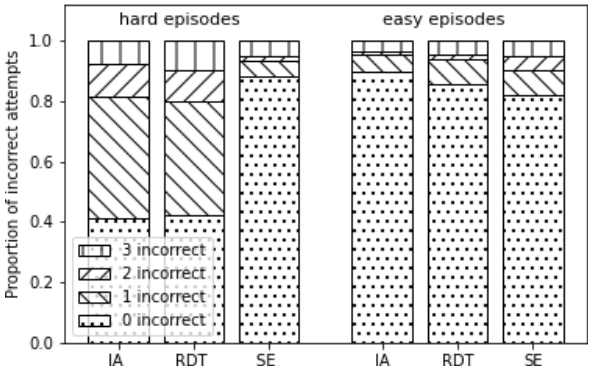


Figure 3: Incorrect choices made for each algorithm.

the mean age was 29.95 (5 did not report). 28 runs were collected for IA, 30 for RDT and 32 for SE. Each run consists of 12 episodes as described in the previous section. We removed single episodes from the data when the gripper stood still for 20 seconds or longer, assuming that the participant had abandoned it, resulting in a total of 331 episodes for IA, 351 episodes for RDT and 384 episodes for SE.

The results are summarized in Tables 2 and 3. All three algorithms achieve similarly high success rates overall as well as for the *easy* episodes. In the *hard* episodes, SE performs best. Participants interacting with IA were fastest in all settings and overall slowest with SE. Figure 3 additionally shows a more detailed visualization of failed attempts for each algorithm and setting.

The most striking differences between the three methods can be seen when looking at the number of failed attempts and the number of feedback messages participants received. Participants had a maximum of 3 attempts to identify the target object before they would see the next, unrelated episode. Overall, participants needed about half as many attempts with SE (0.29) compared with RDT (0.58) and clearly fewer than IA (0.52). However, this differed distinctly when separating easy and hard episodes. In easy episodes, participants with IA needed fewer attempts than both RDT and SE. In hard episodes, participants needed more than three times as many attempts with both IA and RDT as with SE. Unsurprisingly, participants received many more feedback messages with SE in all settings since SE evaluates the position of the gripper continuously rather than only after a certain distance (cf. Section 3.3).

Each participant interacted with only one algorithm, which makes direct comparison impossible. Instead, we asked for their subjective ratings as summarized in Table 3. All algorithms' output was rated as rather machine-like. Despite the underspecified instructions and high number of feedback messages, SE was rated as most competent, likeable, and friendly, with IA close in scores. None of the score averages surpass 5 on a scale up to 7, so a lot of room for improvement exists. Note that many factors can influence this rating, including the particular voice and feedback verbalization.

Based on these results, movement and timing information are appropriate indicators of the IF's reference resolution in this particular domain. Even without testing different timing settings, the success rate is high for all settings, showing that the instructions and feedback are interpretable. The results also give an indication on the level of performance we can expect in easy vs. hard settings and serve as a baseline for comparison when training a model using raw image data as input. The success rate when considering only the participants' first guesses differs greatly between these two settings for the IA and RDT algorithms; both achieve about twice the success rate in easy vs. hard configurations.

5 Conclusion

We have shown how rule-based REG algorithms can be enhanced with timing- and movement-based feedback to increase referential success, especially in ambiguous configurations, and without having to generate spatial relations between objects. The success rates give us a baseline for generating such RE based on raw image data, without access to absolute property values. As seen in Figure 3, these baselines differ for the three algorithms depending on the particular configuration of objects. For unambiguous settings, instructions given by all algorithms were picked on first try in most of the cases, while the success rate dropped visibly for ambiguous settings when IA and RDT gave an instruction. This is important for using these instructions as input for other learning mechanisms.

Our tool (mentioned in Section 4.1) lets us easily convert the symbolic visual game boards into images, making it suitable to compare the exact same settings with neural network models and generating the necessary amount of unbiased object configurations as training data. Instead of letting human annotators formulate instructions that potentially vary significantly in their verbalizations, we will use the rule-based algorithms to generate the training instructions we have tested with users in this paper.

6 Limitations

We acknowledge that the sample of participants is small and there is no guarantee that participants have focussed on the task at all times. We have removed outliers where the gripper stayed idle for a long time as explained in Section 4 but participants carried out the interaction in the environment of their choice rather than in the lab where they could have been supervised. Only 5% of participants reported to be native speakers of English. The remainder self-reported a mean fluency of 5.26 on a scale from 1 ("limited fluency") to 7 ("full fluency").

Acknowledgements

This work was partially funded by DFG project 423217434 ("recolage").

References

Luciana Benotti and Patrick Blackburn. 2021. Grounding as a Collaborative Process. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 515–531, Online. Association for Computational Linguistics.

Donna Byron, Alexander Koller, Jon Oberlander, Laura Stoia, and Kristina Striegnitz. 2007. Generating Instructions in Virtual Environments (GIVE): A Challenge and an Evaluation Testbed for NLG. In *Proceedings of the Workshop on Shared Tasks and Comparative Evaluation in Natural Language Generation*.

Herbert H. Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1–39.

Robert Dale and Ehud Reiter. 1995. Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19(2):233–263.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Jose M. F. Moura, Devi Parikh, and Dhruv Batra. 2017. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335.

Alexandre Denis. 2010. Generating Referring Expressions with Reference Domain Theory. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics.

Nikos Engonopoulos, Martin Villalba, Ivan Titov, and Alexander Koller. 2013. Predicting the Resolution of Referring Expressions from User Behavior. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1354–1359. Association for Computational Linguistics.

Peter A. Heeman and Graeme Hirst. 1995. Collaborating on Referring Expressions. *Computational Linguistics*, 21(3):351–382.

Nikolina Koleva, Martin Villalba, Maria Staudte, and Alexander Koller. 2015. The Impact of Listener Gaze on Predicting Reference Resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 812–817, Beijing, China. Association for Computational Linguistics.

Alexander Koller, Maria Staudte, Konstantina Garoufi, and Matthew Crocker. 2012. Enhancing Referential Success by Tracking Hearer Gaze. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '12, pages 30–39, Seoul, South Korea. Association for Computational Linguistics.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. 2019.

Habitat: A Platform for Embodied AI Research. *arXiv:1904.01201 [cs]*.

Maria Staudte, Alexander Koller, Konstantina Garoufi, and Matthew Crocker. 2012. Using listener gaze to augment speech generation in a virtual 3D environment. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 34.

Kristina Striegnitz, Hendrik Buschmeier, and Stefan Kopp. 2012. Referring in Installments: A Corpus Study of Spoken Object References in an Interactive Virtual Environment. In *INLG 2012 Proceedings of the Seventh International Natural Language Generation Conference*, pages 12–16, Utica, IL. Association for Computational Linguistics.

Michael K. Tanenhaus, Michael J. Spivey-Knowlton, Kathleen M. Eberhard, and Julie C. Sedivy. 1995. Integration of visual and linguistic information in spoken language comprehension. *Science (New York, N.Y.)*, 268(5217):1632–1634.

Kees van Deemter, Albert Gatt, Ielka van der Sluis, and Richard Power. 2012. Generation of Referring Expressions: Assessing the Incremental Algorithm. *Cognitive Science*, 36(5):799–836.

Sina Zarrieß, Julian Hough, Casey Kennington, Ramesh Manuvinakurike, David DeVault, Raquel Fernández, and David Schlangen. 2016. PentoRef: A Corpus of Spoken References in Task-oriented Dialogues. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 125–131, Portorož, Slovenia. European Language Resources Association (ELRA).

Sina Zarrieß and David Schlangen. 2018. Being data-driven is not enough: Revisiting interactive instruction giving as a challenge for NLG. In *Proceedings of the Workshop on NLG for Human–Robot Interaction*, pages 27–31, Tilburg, The Netherlands. Association for Computational Linguistics.

A Material

Figures 4 and 5 shows example episodes. Figure 6 shows the initial screen that participants saw when starting the data collection interface. Demographic questions in the post-task questionnaire were voluntary.

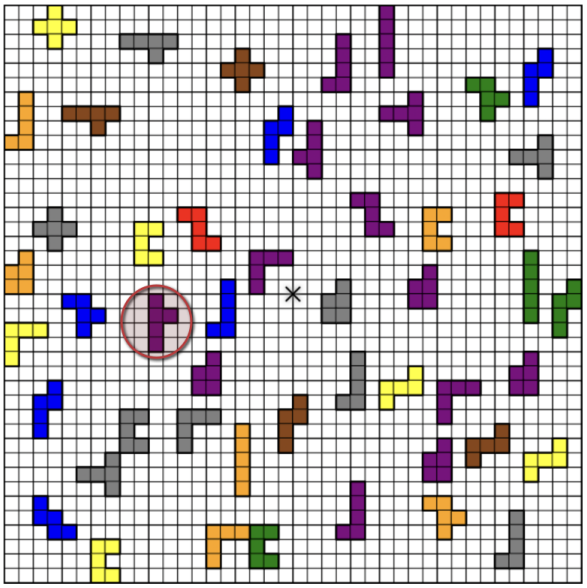


Figure 4: An example *easy* episode. The target object is circled, the *gripper* is positioned in the center.

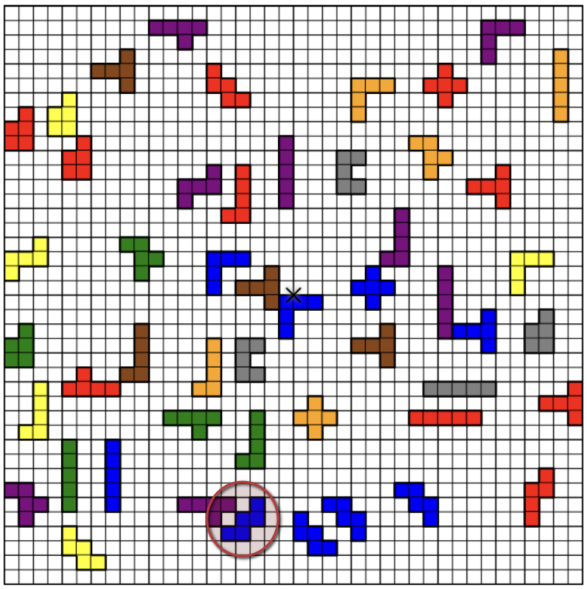


Figure 5: Example of a *hard* episode. The target object is circled, the *gripper* is positioned in the center.

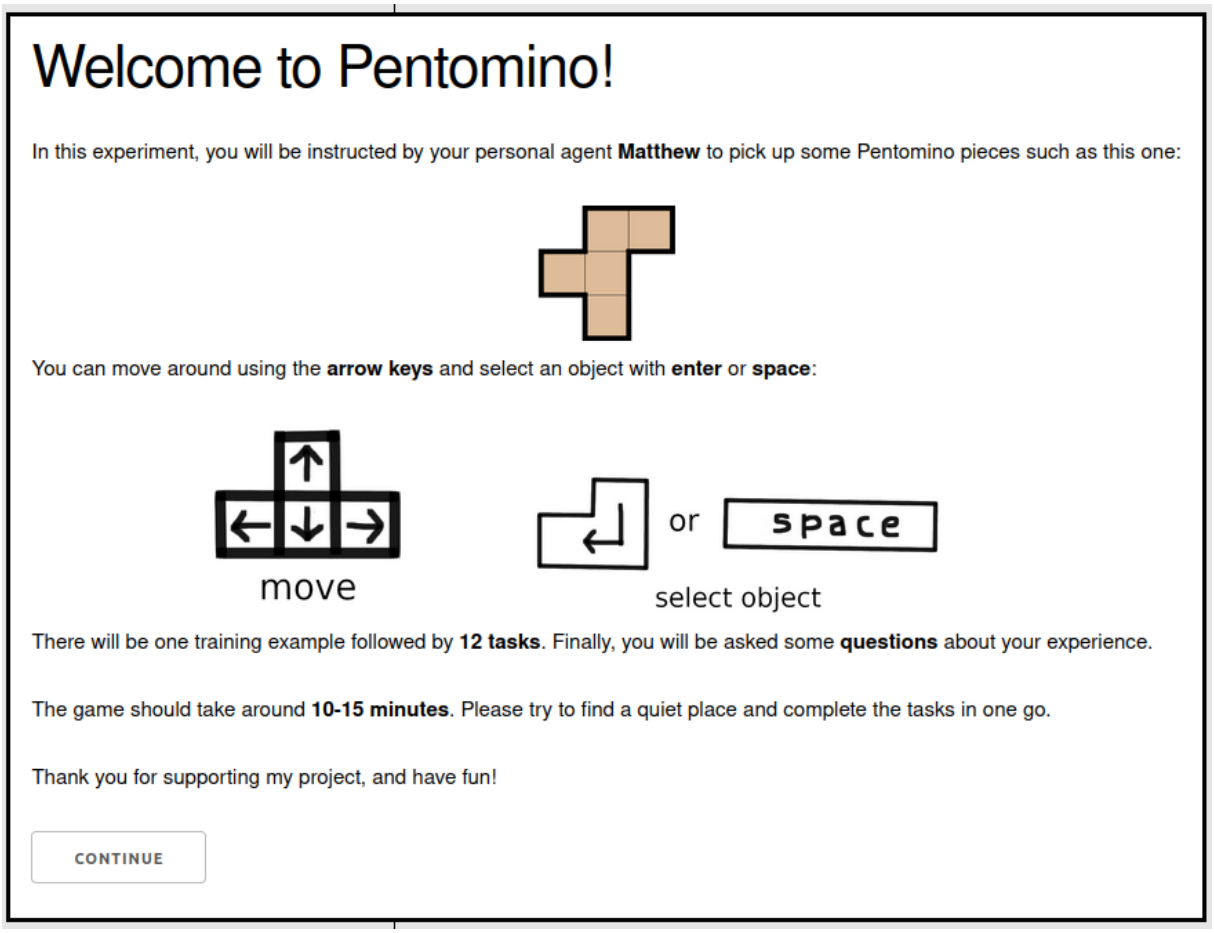


Figure 6: The welcome screen of the data collection.