Language-Conditioned Object Detection and Manipulation

Ayush Jain

CMU-RI-TR-23-69

Aug 22, 2023

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Submitted in partial fulfillment of the requirements for the degree of Masters in Robotics.

To my loving sister, my partner in crime, Dr Apurva Jain

Abstract

Traditional object detection methods are often confined to predefined object vocabularies, limiting their versatility in real-world scenarios where robots need to understand and execute diverse household tasks. Additionally, the 2D and 3D perception communities have typically pursued separate approaches tailored to their respective domains.

In this thesis, we present a language-conditioned object detector with an open and adaptable vocabulary, capable of seamlessly operating in both 2D and 3D environments with minimal architectural adjustments. Our detector incorporates top-down guidance from language commands to direct its attention within the visual stream, while also leveraging bottom-up information from pre-trained object detectors. We demonstrate its state-of-the-art performance in both 2D and 3D contexts on widely-recognized benchmarks.

Furthermore, we showcase its practical utility in language-guided robot manipulation. Central to our model are energy-based concept generation modules, proficient in handling longer instructions and novel spatial concept combinations. We evaluate our model on established instruction-guided manipulation benchmarks, including newly introduced benchmarks for compositional instructions. Notably, our model demonstrates the ability to execute highly compositional instructions zero-shot in both simulation and real-world settings.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Prof. Katerina Fragkiadaki, for taking me under her wing when I was a clueless undergraduate student and nurturing me into whatever researcher I am today. She instilled in me the importance of tackling challenging and impactful questions, and I will never forget her words: "It is better to keep failing on the right problem than succeeding on the wrong problem." This quote has been my guiding principle, especially during moments when experiments didn't yield the desired results. Katerina taught me to be courageous, to think deeply, and to continually expand my knowledge. Her infectious energy and enthusiasm have been a constant source of motivation. I am immensely grateful not only for her guidance as a researcher but also for her support during a personal loss. Katerina is someone I truly look up to, and I eagerly anticipate continuing to learn from her in the future.

I would like to express my deepest gratitude to Tom M. Mitchell for his unwavering support and enlightening discussions. Attending your group meetings was always a delightful and intellectually stimulating experience. Beyond your contributions as a researcher, you inspire me to strive for excellence not only in my academic pursuits but also in becoming a better human being. If I can even become 1% of the remarkable person you are, I would consider my life a resounding success. Thank you for everything, and I eagerly anticipate our continued interactions and collaborations in the years ahead.

I would also like to extend my thanks to Shubham Tulsiani. While our interactions may not have been as extensive as I had hoped, the moments we did spend together during part-based project meetings and in your 3D vision course were truly awesome. I look forward to the opportunity of getting to know you better and learning from you more closely in the coming years.

I would like to express my heartfelt appreciation to Nikos for his invaluable presence as a friend, mentor, and collaborator throughout the past three years. Working alongside you has not only been productive but also incredibly enjoyable. Nikos, you have consistently been there for me during moments of both joy and sorrow, offering unwavering support without any hint of judgment. Sitting beside you and engaging in collaborative endeavors have been the highlights of my master's journey, and I eagerly look forward to our continued collaboration in the future.

I extend heartfelt appreciation to Adam Harley for his invaluable contributions over the past few years. Adam possesses the wisdom of a sage – always calm, composed, and offering profound insights. Additionally, his compassion and friendly nature have made working with him an enlightening experience. Whenever I encountered challenges, Adam provided sage advice that has repeatedly proven invaluable. I will forever cherish the memories of our journey to Jerusalem during ECCV 2022 and the wisdom you shared with me during that time. Presently, I thoroughly enjoy our collaborations and weekly meetings, and I hope our tradition of finding one bug per week remains fruitful. I have no doubt that Adam will become an exceptional advisor, and his future students will be fortunate to have him as their mentor.

I would also like to extend my gratitude to my collaborators and friends, Gabe Sarch and Mayank Singh. Working with them has been truly wonderful, but the moments we've spent partying together have been, well, let's just say interesting. As the sole person in our group who doesn't partake in drinking, I've had the unique opportunity to witness and remember all the unforgettable nights we've shared. These moments will forever hold a special place in my heart, even if some details might be a bit hazy for the others. Who knew being the designated sober friend could provide so much entertainment? I wish Gabe an abundant supply of Takis, and Mayank, may you continue your impressive career as a professional table-smasher!

During my time at CMU, I had the privilege of working with a remarkable group of individuals. I would like to express my sincere gratitude to Aishwarya Jadhav, Andy Fang, Fish Tung, Ishita Mediratta, Mihir Prabhudesai, Pushkal Katara, Shamit Lal, Theophile Gervet, Tsung-Wei Ke, Wen-Hsuan Chu, Yunchu Zhang, and Zhou Xian. Your collaboration and support have been instrumental in making this thesis possible. Thank you all for your invaluable contributions.

Lastly, I want to express my deepest gratitude to my parents, Abha and Ajay Kumar Jain, for their unwavering support. They shielded me from difficulties, encouraged me to pursue my dreams, and inspired me to keep pushing forward, even during uncertain times. I would like to thank my sister, Dr. Apurva Jain, for always being there for me through

thick and thin. Without you, I wouldn't be where I am today. Your encouragement during moments of doubt prevented me from settling for a job as a software developer. Although living without you is an immense challenge, a part of me will forever carry your spirit and brilliance. It is with great honor and love that I dedicate this thesis to her, as a tribute to her unwavering belief in me and her everlasting impact on my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Object detection is the fundamental computer vision task of finding all "objects" that are present in a visual scene. However, this raises the question, what is an object? Typically, this question is side-stepped by defining a vocabulary of categories and then training a model to detect instances of this vocabulary. This means that if "apple" is not in this vocabulary, the model does not consider it as an object. The problem gets even worse when we try to integrate these object detectors into real household agents. Imagine that we want a robot that can pick up "your favorite green mug from the table right in front of you". We want the robot to specifically detect the "green mug" which is on the "table in front of you" and not any other mug or table. Obviously, treating descriptions such as "green mug from the table right in front of you" as separate classes in the detector's vocabulary cannot scale; one can come up with countless variations of such descriptions.

To address this problem, In Chapter 2 of this thesis, we introduce Bottom-up Top-Down DEtection TRansformer (BUTD-DETR pron. Beauty-DETER), a model that conditions directly on a language utterance and detects all objects that the utterance mentions. When the utterance is a list of object categories, BUTD-DETR operates as a standard object detector. It is trained from both fixed vocabulary object detection datasets and referential grounding datasets which provide image-language pairs annotated with the bounding boxes for all objects referred to in the language utterance. With minimal changes, BUTD-DETR grounds language phrases both in 3D point clouds and 2D images. Thus, we relax the assumption of closed-vocabulary object detectors and enable our model to detect objects mentioned in free-flowing natural language.

Chapter 3 introduces our approach to scene re-arrangement, where we manipulate a diverse range of objects to fulfill specific language instructions. We break down this task into three key components: perception, concept learning, and low-level manipulation. To begin with, we utilize the open-vocabulary detector introduced in Chapter 2 for perceiving the objects referenced in the given sentence. This detector identifies all relevant objects mentioned in the instruction. Next, for concept learning, we establish a library of energy-based concept learner models, each representing a spatial relation (e.g., left, right, above, below, inside, etc.). At test time, these models can be combined in a zero-shot manner to perform compositional tasks like "Place the strawberry to the left of the banana and to the right of the cherry." by simply jointly optimizing over sum of energies from the "left" and the "right" concept modules. The energy-based models operate on the abstraction of bounding boxes predicted by the object detector and generate new "goal" locations for each object, ensuring the fulfilment of the language goal. Finally, we design low-level manipulation policies that utilize the pick and place locations generated by the object detector and the energy-based models, respectively, to execute the actual manipulation. Through our experiments, we demonstrate that our proposed models outperform existing state-of-the-art approaches in language-guided instruction manipulation scenarios both in simulation and real-world. Notably, they exhibit exceptional out-of-domain generalization capabilities, effectively handling novel objects, colors, and backgrounds, surpassing existing state-of-the-art approaches.

Finally, Chapter 4 provides a summary of these findings and delves into future research prospects.

Chapter 2

Bottom Up Top Down Detection Transformers for Language Grounding in Images and Point Clouds

2.1 Introduction

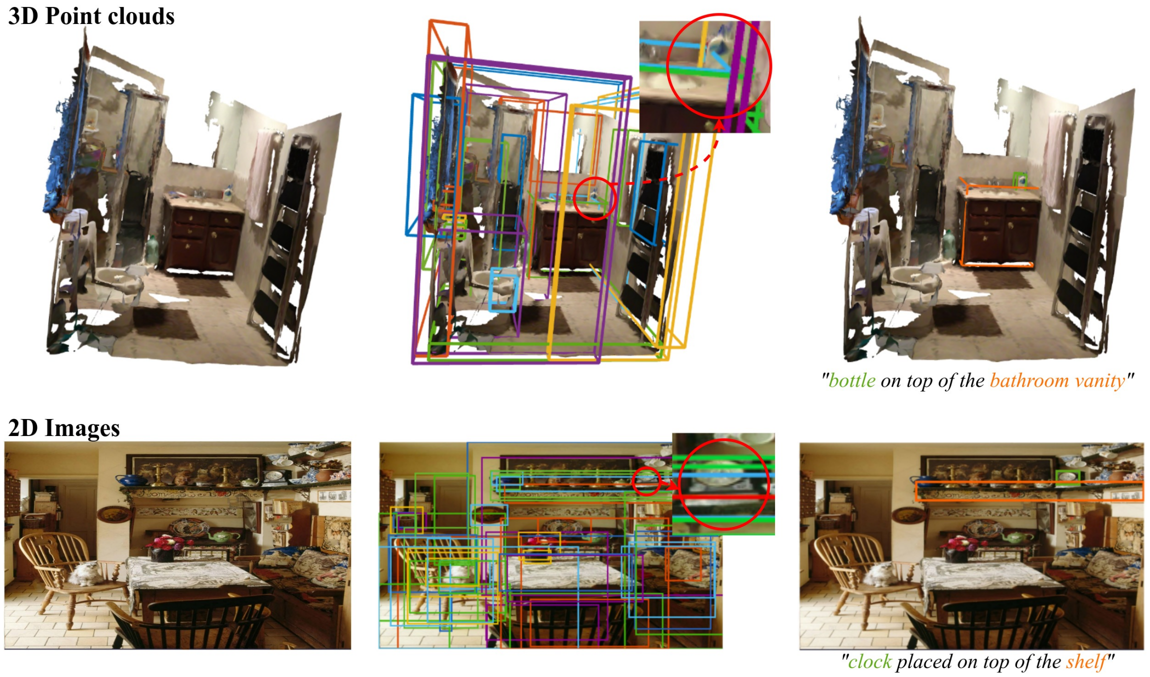

3D Point clouds
"bottle on top of the bathroom vanity"
2D Images
"clock placed on top of the shelf"

Figure 2.1: **Language-modulated 3D (*top*) and 2D (*bottom*) detection with BUTD-DETR.** *Middle:* State-of-the-art object detectors often fail to localize small, occluded or rare objects (here they miss the clock on the shelf and the bottle on the cabinet). *Right:* Language-driven and objectness-driven attention in BUTD-DETR modulates the visual processing depending on the referential expression while taking into account salient, bottom-up detected objects, and correctly localizes all referenced objects.

Language-directed attention helps us localize objects that our "bottom-up", task-agnostic perception may miss. Consider Fig. 2.1. The utterance *"bottle on top of the bathroom vanity"* suffices to direct our attention to the reference object, even though it is far from salient. Language-directed perception adapts the visual processing of the input scene according to the utterance. Object detectors instead apply the same computation in each scene, which can miss task-relevant objects.

Most existing language grounding models use object proposal bottlenecks: they select the referenced object from a pool of object proposals provided by the pre-trained

This chapter is based on the paper previously published at ECCV 2022 [36] object detector [17, 19, 30, 37, 44]. This means they cannot recover objects or parts that a bottom-up detector misses. This is limiting since small, occluded, or rare objects are hard to detect without task-driven guidance. For example, in Figure 2.1 middle, state-of-the-art 2D [84] and 3D [61] detectors miss the clock on the shelf and the bottle on the bathroom vanity, respectively.

Recently, Kamath et al. [42] introduced MDETR, a language grounding model for 2D images that decodes object boxes using a DETR [5] detection head and aligns them to the relevant spans in the input utterance, it does not select the answer from a box proposal pool. The visual computation is modulated based on the input utterance through several layers of self-attention on a concatenation of language and visual features. MDETR achieves big leaps in performance in 2D language grounding over previous box-bottlenecked methods.

We propose a model for grounding referential utterances in 3D and 2D visual scenes that builds upon MDETR, which we call BUTD-DETR (pronounced Beauty-DETR), as it uses both box proposals, obtained by a pre-trained detector "bottom-up" and "top-down" guidance from the language utterance, to localize the relevant objects in the scene. BUTD-DETR uses box proposals obtained by a pre-trained detector as an additional input stream to attend on; however, it is not box-bottlenecked and still decodes objects with a detection head, instead of selecting them from the input box stream. Current object detectors provide a noisy tokenization of the input visual scene that, as our experiments show, is a useful cue to attend on for multimodal reasoning. Second, BUTD-DETR augments grounding annotations by configuring annotations for object detection as detection prompts to be grounded in visual scenes. A detection prompt is a list of object category labels, e.g., *"Chair. Door. Person. Bed."*. We train the model to ground detection prompts by localizing the labels that are present in the image and learn to discard labels that are mentioned but do not correspond to any objects in the scene. Third, BUTD-DETR considers improved bounding box - word span alignment losses that reduce noise during alignment of object boxes to noun phrases in the referential utterance.

We test BUTD-DETR on the 3D benchmarks of [2, 6] and 2D benchmarks of [45, 107]. In 3D point clouds, we set new state-of-the-art in the two benchmarks of Referit3D [2] and ScanRefer [6] and report significant performance boosts over all prior methods (12.6% on SR3D, 11.6% on NR3D and 6.3% on ScanRefer), as well as over a direct MDETR-3D implementation of ours that does not use a box proposal stream or detection prompts during training. In 2D images, our model obtains competitive performance with MDETR on RefCOCO, RefCOCO+ and Flickr30k, and requires less than half of the GPU training time due to the cheaper deformable attention in the visual stream. We ablate each of the design choices of the model to quantify their contribution to performance.

In summary, our contributions are: **(i)** A model with SOTA performance across both 2D and 3D scenes with minor changes showing that modulated detection in 2D images can also work in 3D point clouds with appropriate visual encoder and decoder modifications. **(ii)** Augmenting supervision with detection prompts, attention on an additional input box stream and improved bounding box - word span alignment losses. **(iii)** Extensive ablations to quantify the contribution of different components of our model. We make our code publicly available at `https://butd-detr.github.io`.

2.2 Related work

Object detection with transformers

Object detectors are trained to localize all instances of a closed set of object category labels in images and 3D point-clouds. While earlier architectures pool features within proposed boxes to decode objects and classify them into categories [29, 57, 83], recent methods pioneered by DETR [5] use transformer architectures where a set of object query vectors attend to the scene and among themselves to decode object boxes and their labels. DETR suffers from the quadratic cost of within image features self attention. D(eformable)-DETR [111] proposes deformable attention, a locally adaptive kernel that is predicted directly in each pixel location without attention to other pixel locations, thus saving the quadratic cost of pixel-to-pixel attention. Our model builds upon deformable attention for feature extraction from RGB images. [61, 71] extend detection transformers to 3D point cloud input.

2D referential language grounding

Referential language grounding [45] is the task of localizing the object(s) referenced in a language utterance. Most 2D language grounding models obtain sets of object proposals using pre-trained object detectors and the original image is discarded upon extraction of the object proposals [17, 19, 30, 37, 44]. Many of these approaches use multiple layers of attention to fuse information across both, the extracted boxes and language utterance [8, 62, 105]. Recently, a few approaches directly regress the target bounding box without using pre-trained object proposals. In [7] language and visual features cross-attend and are concatenated to predict the box of the referential object. Yang et al. [103] extends the YOLO detector [83] to referential grounding by channel-wise concatenating language, visual and spatial feature maps and then regressing a single box using the YOLO box prediction head. [87] performs a fusion similar to [103], then selects a single box from a set of anchor boxes and predicts a deformation of it, much like the Faster-RCNN object detector [84]. While previous approaches encode the whole text input into a single feature vector, [104] further improves performance by recursively attending on different parts of the referential utterance. Lastly, [12] encodes the image and utterance with within- and cross-modality transformers, and a special learnable token regresses a single box. In contrast to our method, all these works predict a single bounding box per image-utterance pair. Our work builds upon MDETR of Kamath et al. [42] that modulates visual processing through attention to the input language utterance and decodes objects from queries similar to DETR, without selecting from a pool of proposals. Both our method and MDETR can predict multiple instances being referred to, as well as ground intermediate noun phrases. Concurrent to our work, GLIP [49] shows that adding supervision from detection annotations can improve 2D referential grounding. Our work independently confirms this hypothesis in 2D and also shows its applicability on the 3D domain.

3D referential language grounding

has only recently gained popularity [2, 6]. To the best of our knowledge, all related approaches are box-bottlenecked: they extract 3D object proposals and select one as their answer. Their pipeline can be decomposed into three main steps: i) Representation of object boxes as point features [105], segmentation masks [108] or pure spatial/categorical features [86]. ii) Encoding of language utterance using word embeddings [86, 105] and/or scene graphs [18]. iii) Fusion of the two modalities and scoring of each proposal using graph networks [31] or Transformers [105]. Most of these works also employ domain-specific design choices by explicitly encoding pairwise relationships [25, 31, 108] or by relying on heuristics, such as restricting attention to be local [108, 110] and ignoring input modalities [86]. Such design prevents those architectures from being applicable to both the 3D and 2D domains simultaneously.

Due to the inferior performance of 3D object detectors in comparison to their 2D counterparts, popular benchmarks for 3D language grounding, such as Referit3D [2] provide access to ground-truth object boxes at test time. The proposed BUTD-DETR is the first 3D language grounding model that is evaluated on this benchmark without access to oracle 3D object boxes.

2.3 Method

We first describe MDETR [42] in Section 2.3.1. Then, we present BUTD-DETR's architecture in Section 2.3.2, supervision augmentation with detection prompts in Section 2.3.3 and its training objectives in Section 2.3.4.

2.3.1 Background: MDETR

MDETR is a 2D language grounding model that takes a referential utterance and an RGB image as input and localises in the image all objects mentioned in the utterance. MDETR encodes the image with a convolutional network [27] and the language utterance with a RoBERTa encoder [59]. It then fuses information across the language and visual features through multiple layers of self-attention on the concatenated visual and language feature sequences. In MDETR's decoder, a set of query vectors iteratively attend to the contextualized visual features and self-attend to one another, similar to the DETR's [5] decoder. Finally, each query decodes a bounding box and a confidence score over each word in the input utterance, which associates the box to a text span.

The predicted boxes are assigned to ground-truth ones using a Hungarian matching, similar to [5]. Upon matching, the following losses are computed:

- A bounding box loss between predicted boxes and the corresponding ground-truth ones. This is a combination of L1 and generalized IoU [85] losses.
- A soft token prediction loss. A query matched to a ground-truth box is trained to decode a uniform distribution over the language token positions that refer to that object. Queries not matched to ground-truth targets are trained to predict a no-object label.
- Two contrastive losses between query and language token features. The first one, called *object contrastive loss*, pulls an object query's features closer to the features of the corresponding ground-truth span's word tokens, and further than all other tokens. The second one, called *token contrastive loss*, pulls the features of a ground-truth span's token closer to the corresponding object query features, and further than all other queries.

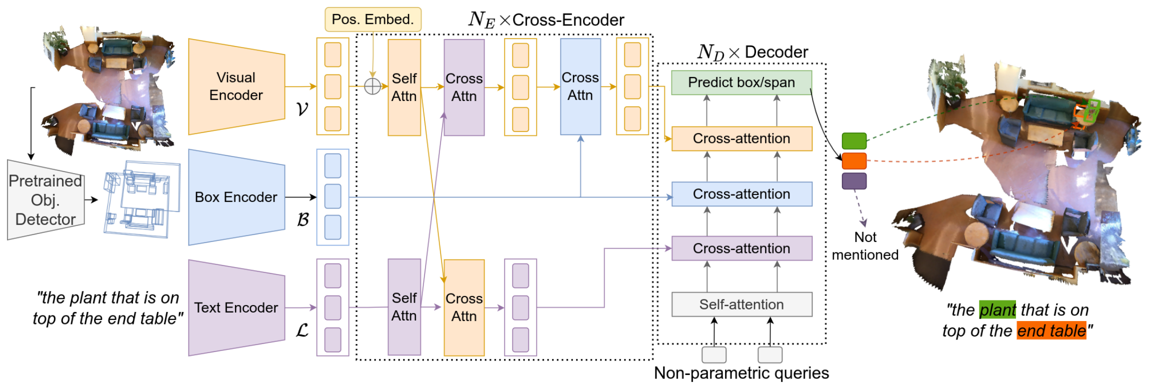


Figure 2.2: **BUTD-DETR architecture.** Given a visual scene and a referential utterance, the model localizes all object instances mentioned in the utterance. A pre-trained object detector extracts object box proposals. The visual scene features, the language utterance and the labelled box proposals are encoded into corresponding sequences of visual, word and box tokens using visual, language and box encoders, respectively. The three streams cross-attend and finally decode boxes and corresponding spans in the language utterance that each decoded box refers to. We visualize here the model operating on a 3D point cloud; an analogous architecture is used for 2D image grounding.

2.3.2 Bottom-up Top-down DETR (BUTD-DETR)

The architecture of BUTD-DETR is illustrated in Figure 2.2. Given a referential language utterance, e.g., “find the plant that is on top of the end table” and a visual scene, which can be a 3D point cloud or a 2D image, BUTD-DETR is trained to localize all objects mentioned in the utterance. In the previous example, we expect one box for the “plant” and one for the “end table”. The model attends across image/point cloud, language and box proposal streams, then decodes the relevant objects and aligns them to input language spans.

Within-modality encoder

In 2D, we encode an RGB image using a pre-trained ResNet101 backbone [26]. The 2D appearance visual features are added to 2D Fourier positional encodings, same as in [35, 111]. In 3D, we encode a 3D point cloud using a PointNet++ backbone [78]. The 3D point visual features are added to learnable 3D positional encodings, same as in [61]: we pass the coordinates of the points through a small multilayer perceptron

(MLP). Let $\mathcal{V} \in \mathbb{R}^{n_v \times c_v}$ denote the visual token sequence, where n_v is the number of visual tokens and c_v is the number of visual feature channels.

The words of the input utterance are encoded using a pre-trained RoBERTa [59] backbone. Let $\mathcal{L} \in \mathbb{R}^{n_\ell \times c_\ell}$ denote the word token sequence.

A pre-trained detector is used to obtain 2D or 3D object box proposals. Following prior literature, we use Faster-RCNN [84] for RGB images, pre-trained on 1601 object categories of Visual Genome [47], and Group-Free detector [61] for 3D point clouds pre-trained on a vocabulary of 485 object categories on ScanNet [11]. The detected box proposals that surpass a confidence threshold are encoded using a box proposal encoder, by mapping their spatial coordinates and categorical class information to an embedding vector each, and concatenating them to form an object proposal token. We use a pre-trained and frozen RoBERTa [59] backbone to encode the semantic categories of proposed boxes. Let $\mathcal{O} \in \mathbb{R}^{n_o \times c_o}$ denote the object token sequence.

The 3D detector is trained on ScanNet and all 3D benchmarks we use are also ScanNet-based. This creates a discrepancy in the quality of the detector's predictions between train and test time, as it is far more accurate on the training set. As a result, we find that BUTD-DETR tends to rely on the detector at training time and generalizes less at test time, where the detector's predictions are much noisier. To mitigate this, we randomly replace 30% of the detected boxes at training time with random ones. This augmentation leads to stronger generalization when the detector fails to locate the target object. Note that this is not the case in 2D, where the detector is trained on a different dataset.

All visual, word and box proposal tokens are mapped using (different per modality) MLPs to same-length feature vectors.

Cross-modality Encoder

The visual, language and box proposals, interact through a sequence of N_E cross-attention layers. In each encoding layer, visual and language tokens cross-attend to one another and are updated using standard key-value attention. Then, the resulting language-conditioned visual tokens attend to the box proposal tokens. We use standard attention for both streams in 3D and deformable attention [111] for the visual stream in 2D.

In contrast to MDETR, BUTD-DETR keeps visual, language and box stream separate in the encoder instead of concatenating them. This enables us to employ deformable attention [111] in self and cross attention layers involving the visual stream in 2D domain. Deformable attention involves computing bilinearly interpolated features which is expensive and non-robust in discontinous and sparse modalities like pointclouds, hence we use vanilla attention in 3D. In our experiments, we show that concatenation versus keeping separate streams performs similarly in 3D referential grounding.

Decoder

BUTD-DETR decodes objects from contextualized features using non-parametric queries in both 2D and 3D, similar to [61, 111]. Non-parametric queries are predicted by visual tokens from the current scene, in contrast to parametric queries used in DETR [5] and MDETR [42] that correspond to a learned set of vectors shared across all scenes. Specifically, the contextualized visual tokens from the last multi-modality encoding layer predict confidence scores, one per visual token. The top-K highest scoring tokens are each fed into an MLP to predict a vector which stands for an *object query*, i.e., a vector that will decode a box center and size relative to the location of the corresponding visual token, similar to D-DETR [111]. The query vectors are updated in a residual manner through N_D decoder layers. In each decoder layer, we employ four types of attention operations. First, the queries self-attend to one another to contextually refine their estimates. Second, they attend to the contextualized word embeddings to condition on the language utterance. Next, they attend to the box proposal tokens and then in the image or point visual tokens. At the end of each decoding layer, there is a prediction head that predicts a box center displacement, height and width vector, and a token span for each object query that localizes the corresponding object box and aligns it with the language input.

2.3.3 Augmenting supervision with detection prompts

Object detection is an instance of referential language grounding in which the utterance is a single word, namely, the object category label. Language grounding models have effectively combined supervision across referential grounding, caption description

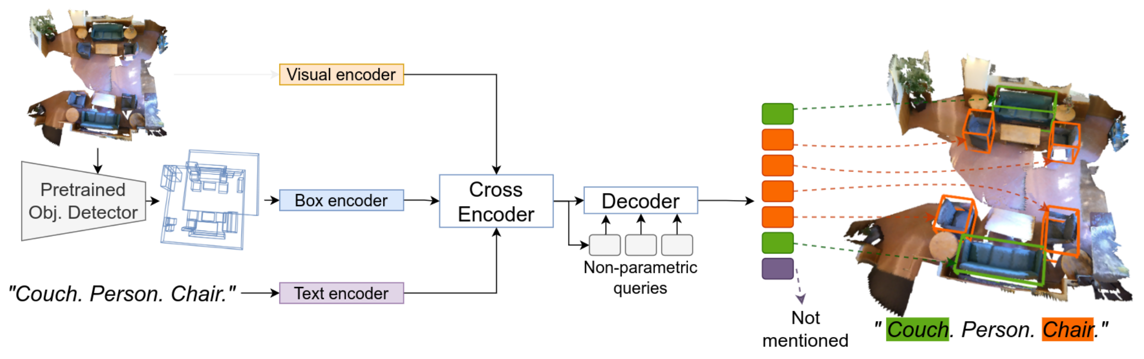


Figure 2.3: **Augmenting referential grounding supervision with detection prompts.** A detection prompt is constructed by sequencing sampled object category labels (here *couch*, *person* and *chair*). The task is to localize all instances of mentioned objects and associate them with the correct span in the prompt. 50% of the sampled labels are negative, i.e., they have no corresponding object instance in the scene. The model learns not to associate these spans with predicted boxes.

and question answering tasks [62, 63], which is an important factor for their success. Object detection annotations have not been considered so far as candidates for such co-training.

We cast object detection as grounding of detection prompts, namely, referential utterances comprised of a list of object category labels, as shown in Figure 2.3. Specifically, given the detector's vocabulary of object category labels, we randomly sample a fixed number of them—some appear in the visual scene and some do not—and generate synthetic utterances by sequencing the sampled labels, e.g., *"Couch. Person. Chair. Fridge."*, we call them detection prompts. We treat these prompts as referential utterances to be grounded: the task is to localize *all* object instances of the category labels mentioned in the prompt if they appear in the scene. The sampling of negative category labels (labels for which there are no object instances present) operates as negative training: the model is trained to not match any boxes to the negative category labels.

2.3.4 Supervision objectives

We supervise the outputs of all prediction heads in each layer of the decoder. We follow MDETR [42] in using Hungarian matching to assign a subset of object queries to the ground-truth object boxes and then compute the bounding box, soft token prediction and contrastive losses. Our bounding box and soft token prediction losses are identical to MDETR's. However, we notice that MDETR's contrastive losses do not compare all object queries and word tokens symmetrically. Specifically, the object contrastive loss supervises only the object queries that are matched to a ground-truth object box. On the other hand, the token contrastive loss includes only the tokens that belong to positive spans, namely, noun phrases with corresponding object instances in the scene. As a result, object queries not matched to any ground-truth object box are not pulled far from non-ground-truth text spans, which means at inference object queries can be close to negative spans. We find this asymmetry to hurt performance, as we show in our experiments.

To address this, we propose a symmetric alternative where the similarities between all object queries and language tokens are considered. We append the span "not-mentioned" to all input utterances. This acts as the ground-truth text span for all object queries that are not assigned to any of the ground-truth objects. The object contrastive loss now supervises all queries and considers the similarities with all tokens. We empirically find that gathering unmatched queries to "not mentioned" is beneficial. This is similar in principle to the soft token prediction loss, where unmatched queries have to predict "no object". In fact, we find that this symmetric contrastive loss is sufficient for our model's supervision, but we observe that co-optimizing for soft token prediction results in faster convergence.

2.4 Experiments

We test BUTD-DETR on grounding referential utterances in 3D point clouds and 2D images. Our experiments aim to answer the following questions:

1. How does BUTD-DETR perform compared to the state-of-the-art in 3D and 2D language grounding?
2. How does BUTD-DETR perform compared to a straightforward extension of the 2D state-of-the-art MDETR [42] model in 3D?
3. How much, if at all, attending to a bottom-up box proposal stream helps performance?
4. How much, if at all, co-training for grounding detection prompts helps performance?
5. How much, if at all, the proposed contrastive loss variant helps performance?

2.4.1 Language grounding in 3D point clouds

We test BUTD-DETR on SR3D, NR3D [2] and ScanRefer [6] benchmarks. All three benchmarks contain pairs of 3D point clouds of indoor scenes from ScanNet [11] and corresponding referential utterances, and the task is to localize the objects referenced in the utterance. The utterances in SR3D are short and synthetic, e.g., *"choose the couch that is underneath the picture"*, while utterances in NR3D and ScanRefer are longer and more natural, e.g. *"from the set of chairs against the wall, the chair farthest from the red wall, in the group of chairs that is closer to the red wall"*. For fair comparison against previous methods, we train BUTD-DETR separately on each of SR3D, NR3D and ScanRefer. We augment supervision in each of the three datasets with ScanNet detection prompts. SR3D provides annotations for all objects mentioned in the utterance, so during training we supervise localization of all objects mentioned. In NR3D and ScanRefer, we use supervision for grounding only the referenced object.

All existing models that have been tested in SR3D or NR3D benchmarks are box-bottlenecked, namely, they are trained to select the answer from a pool of box proposals. They all use **ground-truth 3D object boxes (without category**

Table 2.1: **Results on language grounding in 3D point clouds.** We evaluate top-1 accuracy using ground-truth (`GT`) or detected (`det`) boxes. * denotes method uses extra 2D image features. † denotes evaluation with detected boxes using the authors' code and checkpoints. ‡ denotes re-training using the authors' code. For [110], we compare against their 3D-only version.

	SR3D		NR3D	ScanRefer (Val. Set)	
Method	Acc@0.25(`det`)	Acc.(`GT`)	Acc@0.25(`det`)	Acc@0.25(`det`)	Acc@0.5(`det`)
ReferIt3DNet [2]	27.7†	39.8	24.0†	26.4	16.9
ScanRefer [6]	-	-	-	35.5	22.4
TGNN [31]	-	45.0	-	37.4	29.7
3DRefTransformer [1]	-	47.0	-	-	-
InstanceRefer [108]	31.5‡	48.0	29.9‡	40.2	32.9
FFL-3DOG [18]	-	-	-	41.3	34.0
LanguageRefer [86]	<u>39.5</u>†	56.0	28.6†	-	-
3DVG-Transformer [110]	-	51.4	-	<u>45.9</u>	<u>34.5</u>
TransRefer3D [25]	-	57.4	-	-	-
SAT-2D [105]*	35.4†	<u>57.9</u>	<u>31.7</u>†	44.5	30.1
MDETR-[42]-3D (our impl.)	45.4	-	31.5	47.2	31.9
BUTD-DETR (ours)	**52.1**	**67.0**	**43.3**	**52.2**	**39.8**

labels) as the set of boxes to select from. We thus consider two evaluation setups:

1. `det`: where we re-train previous models using their publicly available code and provide the same 3D box proposals we use in BUTD-DETR, obtained by the Group-Free 3D object detector [61] trained to detect 485 object categories in ScanNet (Section `det` in Table 2.1).
2. `GT`, where we use ground-truth 3D object boxes for our model and baseline (Section `GT` in Table 2.1).

Alongside previous models, we also compare our model against our implementation of the MDETR model in 3D. This is similar to our model but without attention on a box stream, without co-training with detection prompts and with the original contrastive losses proposed by MDETR. We also replace MDETR's parametric object queries with non-parametric one —similar to our model—since they have been shown to be crucial for good performance in 3D [61, 71]. We call this model MDETR-3D. For the sake of completeness, we do have a 3D version of MDETR that uses parametric queries in Table 2.2 and, as expected, it is significantly worse. MDETR does not use a pool of box proposals in any way and hence we cannot report results of MDETR-3D under `GT`.

We show quantitative results of our models against previous works in Table 2.1.

Table 2.2: **Ablation of design choices for BUTD-DETR on SR3D**.

Model	Accuracy
BUTD-DETR	**52.1**
w/o visual tokens	41.9
w/o detection prompts	47.9
w/o box stream	51.0
with MDETR's [42] contrastive loss	49.6
w/o detection prompts; w/o box stream; (MDETR [42]-3D)	45.4
with parametric queries; w/o detection prompts; w/o box stream; (MDETR [42]-3D-Param)	33.8
with concatenated Visual, Language and Object Streams	51.3

We use top-1 accuracy metric, which measures the percentage of times we can find the target box with an IoU higher than the threshold. We report results with IoU@0.25 on SR3D and NR3D; and with both IoU@0.25 and IoU@0.5 on ScanRefer. Please refer to supplementary for more detailed results.

BUTD-DETR outperforms existing approaches as well as MDETR-3D by a large margin under both evaluation setups, `det` and `GT`. It also outperforms the recent SAT-2D [105] that uses additional 2D RGB image features during training. BUTD-DETR does not use 2D image features, but it can be easily extended to do so. We show qualitative results in Figure 2.4.

Ablative analysis

We ablate all our design choices for 3D BUTD-DETR on SR3D benchmark [2] in Table 2.2. We compare BUTD-DETR against the following variants:

- w/o visual tokens: an object-bottlenecked variant, which only attends to the language and box proposal streams and selects one box out of the proposals.
- w/o detection prompts: BUTD-DETR trained solely on SR3D grounding utterances.
- w/o box stream: BUTD-DETR without attention on the box stream.
- w/ MDETR's contrastive loss: BUTD-DETR where we replace our modified contrastive loss with MDETR's.
- w/o detection prompts, w/o box stream, w/ MDETR's contrastive loss: an MDETR [42]-3D implementation.
- w/ parametric queries, w/o detection prompts, w/o box stream, w/ MDETR's

contrastive loss: an MDETR-3D implementation that uses parametric object queries, as in original MDETR.

- w/ concatenated visual, language and box streams: instead of attending to each modality separately, we concatenate the different streams along their sequence dimension.

The conclusions are as follows:

1. **Box bottlenecks hurt:** Models such as BUTD-DETR and MDETR-3D that decode object boxes instead of selecting them from a pool of given object proposals significantly outperform box-bottlenecked variants. BUTD-DETR outperforms by 10.2% an object-bottlenecked variant, that does not attend to 3D point features and does not decode boxes.
2. **BUTD-DETR outperforms MDETR-3D by 6.7%**:
3. **Attention on a box proposal stream helps:** Removing attention on the box stream causes an absolute 1.1% drop in accuracy.
4. **Co-training with detection prompts helps:** Co-training with detection prompts contributes 4.2% in performance (from 47.9% to 52.1%).
5. **BUTD-DETR 's contrastive loss helps:** Replacing our contrastive loss with MDETR's results in drop of 2.5% in absolute accuracy.
6. **Concatenating Visual, Language and Object Streams performs worse than a model that has separate streams for each modality** Our motivation is to keep separate streams in 3D cross-modality encoder and decoder to be consistent with 2D BUTD-DETR as explained in Section 2.3.2. We additionally find that having separate streams gives a boost of 0.8%.

2.4.2 Language grounding in 2D images

We test BUTD-DETR on the referential grounding datasets of RefCOCO [45], RefCOCO+ [107] and Flickr30k entities dataset [76]. We follow the pretrain-then-finetune protocol of MDETR and first pre-train on combined grounding annotations from Flickr30k [76], referring expression datasets [45, 69, 107], Visual Genome [47]. During pre-training the task is to detect all instances of objects mentioned in the utterance.

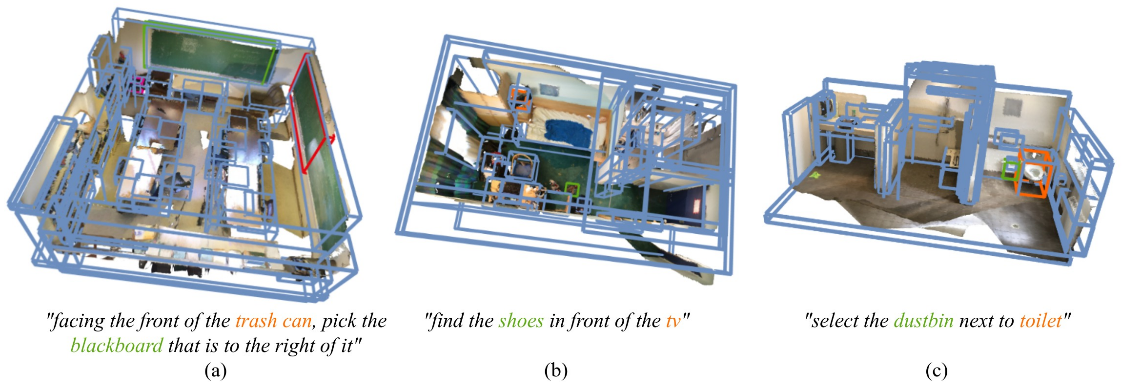

"facing the front of the trash can, pick the blackboard that is to the right of it"
(a)
"find the shoes in front of the tv"
(b)
"select the dustbin next to toilet"
(c)

Figure 2.4: **Qualitative results of BUTD-DETR in the SR3D benchmark**. Predictions for the target are shown in green and for other mentioned objects in orange. The detected proposals appear in blue. (a) The variant without box stream (red box) fails to exploit the information given by the detector, but BUTD-DETR succeeds. (b) The detector misses the "shoes" and any box-bottlenecked variant fails. (c) The detector is successful in finding the "dustbin", still BUTD-DETR refines the box to get a more accurate bounding box.

Different than MDETR, we augment this supervision with detection prompts from the MS-COCO dataset [54]. Following MDETR, we directly evaluate our pre-trained model on Flickr30k without any further fine-tuning and fine-tune for 5 epochs on RefCOCO and RefCOCO+.

We report top-1 accuracy on the standard splits of RefCOCO and RefCOCO+ in Table 2.3 and Recall metric with ANY-BOX protocol [48] on Flickr30k in Table 2.4. Our model and MDETR use the same 200k image-language pairs from COCO [54], Flickr30k [76] and Visual Genome [47]. VisualBERT [48] is trained on COCO captions. UNITER [8] and VILLA [20] use a larger dataset of 4.4M pairs from COCO, Visual Genome, Conceptual-Captions [90], and SBU Captions [74]. In addition, we augment our training set with detection prompts from COCO. BUTD-DETR trains two times faster than MDETR while getting comparable performance. This computational gain comes mostly from deformable attention which is much cheaper than original visual self-attention that scales quadratically with the number of visual tokens, as already reported in [111].

Table 2.3: **Results on language grounding in 2D RefCOCO and RefCOCO+ Datasets on Top-1 accuracy metric using standard splits**. All training times are computed using same V100 GPUs. Training epochs are written as $x + y$ where x = number of pre-training epochs and y = number of fine-tuning epochs. All reported results use ResNet101 backbone.

	RefCOCO			RefCOCO+			Training	Training
Method	val	testA	testB	val	testA	testB	Epochs	GPU Hours
UNITER_L [8]	81.4	87.0	74.2	75.9	81.5	66.7	-	-
VILLA_L [20]	82.4	87.5	74.8	76.2	81.5	66.8	-	-
MDETR [42]	**86.8**	**89.6**	81.4	**79.5**	**84.1**	**70.6**	40 + 5	5560
BUTD-DETR (ours)	85.9	88.5	**81.5**	78.2	82.8	70.0	**12 + 5**	**2748**

Table 2.4: **Results on language grounding in Flickr30k 2D images.** We use Recall@k metric. All training times are computed using same V100 GPUs.

	Val			Test			Training	Training
Method	R@1	R@5	R@10	R@1	R@5	R@10	Epochs	GPU hours
VisualBERT [48]	70.4	84.5	86.3	71.3	85.0	86.5	-	-
MDETR [42]	**82.5**	**92.9**	**94.9**	**83.4**	**93.5**	**95.3**	40	5480
BUTD-DETR (ours)	81.2	90.9	92.8	81.0	91.6	93.2	**12**	**2688**

Ablative analysis

We ablate our model in RefCOCO without pre-training in Table 2.5, since pre-training is computationally expensive due to the size of the combined datasets. Consistent with 3D, removing detection prompts results in an accuracy drop of 2.4%. Additionally removing attention to the box proposal stream results in a drop of 3.1% in accuracy. When replacing our contrastive loss with MDETR's, the model achieves 74.2%, resulting in an additional drop of 2.1% accuracy.

Table 2.5: **Ablation for BUTD-DETR on the RefCOCO validation set**.

Model	Accuracy
BUTD-DETR	**79.4**
w/o det prompts	77.0
w/o box stream w/o det prompts	76.3
w/o box stream w/o det prompts w/ MDETR's [42] contrastive	74.2

2.4.3 Limitations

Our work relies on language-image alignment and does not address how to ground language better and more robustly through abstraction of the visual features, e.g., the fact that *left* and *right* reverse when we change the user's viewpoint, the fact that numbers requires precise counting, or the fact that the *" chair furthest away from the door"* requires to satisfy a logical constraint which our model can totally violate when presented with out-of-distribution visual input. This limitation is a direct avenue for future work.

2.5 Conclusion

We present BUTD-DETR , a model for referential grounding in 3D and 2D scenes, that attends to language, visual and box proposal streams to decode objects mentioned in the referential utterance and align them to corresponding spans in the input. BUTD-DETR builds upon MDETR [42] and outperforms its straightforward MDETR-3D equivalent by a significant margin thanks to attention on labelled bottom-up box proposals, co-training with detection prompts and improved contrastive losses, setting a new state-of-the-art in two 3D language grounding benchmarks. BUTD-DETR is also the first model in 3D referential grounding that operates on the realistic setup of not having access to oracle object boxes, but rather detects them from the input 3D point cloud.

Chapter 3

Energy-based Models are Zero-Shot Planners for Compositional Scene Rearrangement

3.1 Introduction

We consider the scene arrangement task shown in Figure 3.1. Given a visual scene and an instruction regarding object spatial relations, the robot is tasked to rearrange the objects to their instructed configuration. Our focus is on strong generalization to longer instructions with novel predicate compositions, as well as to scene arrangements that involve novel objects and backgrounds.

We propose generating goal scene configurations corresponding to language instructions by minimizing a composition of energy functions over object spatial locations, where each energy function corresponds to a language concept (predicate) in the instruction. We represent each language concept as an n-ary energy function over relative object poses and other static attributes, such as object size. We train these predicate energy functions to optimize object poses starting from randomly sampled object arrangements through Langevin dynamics minimization [16], using a handful of examples of visual scenes paired with single predicate captions. Energy functions

This chapter is based on the paper previously published at RSS 2023 [22]

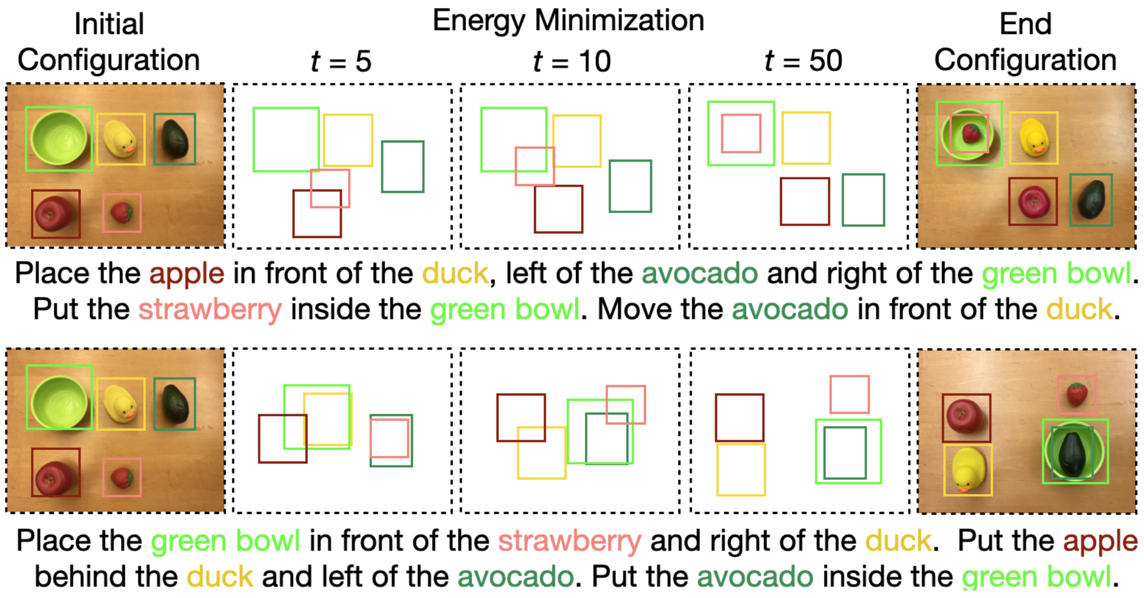

Initial Configuration
Energy Minimization
t = 5
t = 10
t = 50
End Configuration
Place the apple in front of the duck, left of the avocado and right of the green bowl. Put the strawberry inside the green bowl. Move the avocado in front of the duck.
Place the green bowl in front of the strawberry and right of the duck. Put the apple behind the duck and left of the avocado. Put the avocado inside the green bowl.

Figure 3.1: **Energy-based Models are Zero-Shot Planners for Compositional Scene Rearrangement**. We represent language concepts with energy functions over object locations and sizes. Gradient descent on the sum of energy functions, one per predicate in the instruction, iteratively updates the object spatial coordinates and generates a goal scene configuration that satisfies the instruction, if one exists.

can be binary for two-object concepts such as *left of* and *in front of*, or multi-ary for concepts that describe arrangements for sets of objects, such as *line* or *circle*. We show that gradient descent on the sum of predicate energy functions, each one involving different subsets of objects, generates a configuration that jointly satisfies all predicates, if this configuration exists, as shown in Figure 3.1.

We propose a robot learning framework that harnesses minimization of compositions of energy functions to generate instruction-compatible object configurations for robot scene rearrangement. A neural semantic parser is trained to map the input instruction to a set of predicates and corresponding energy functions, and the open-vocabulary visual-language grounding model proposed in Chapter 2 [36] grounds their arguments to objects in the scene, as shown in Figure 3.2. Gradient descent on the sum of energies with respect to the objects' spatial coordinates computes the final object locations that best satisfy the set of spatial constraints expressed in the instruction. Given the predicted object goal locations, we use vision-based pick-and-place policies that condition on the visual patch around the predicted pick and place locations to rearrange the objects [109]. We call our framework Scene Rearrangement via Energy Minimization (SREM).

We test SREM in scene rearrangement of tabletop environments on simulation benchmarks of previous works [91], as well as on new benchmarks we contribute that involve compositional instructions. We curate multiple train and test splits to test out-of-distribution generalization with respect to (i) longer instructions with more predicates, (ii) novel objects and (iii) novel background colors. We show SREM generalizes zero-shot to complex predicate compositions, such as *"put all red blocks in a circle in the plate"* **while trained from single predicate examples**, such as *"an apple inside the plate"* and *"a circle of blocks"*. We show SREM generalizes to real-world scene rearrangement without any fine-tuning, thanks to the object abstractions it operates on. We compare our model against state-of-the-art language-to-action policies [91] as well as Large Language Model planners [33] and show it dramatically outperforms both, especially for long complicated instructions. We ablate each component of our model and evaluate contributions of perception, semantic parsing, goal generation and low-level policy modules to performance.

In summary, our contributions are: **(i)** A novel energy-based object-centric planning framework for zero-shot compositional language-conditioned goal scene generation. **(ii)** A modular system for instruction-guided robot scene rearrangement that uses semantic parsers, vision-language grounding models, energy-based models for scene generation, and vision-based policies for object manipulation. **(iii)** A new instruction-guided scene rearrangement benchmark in simulation with compositional language instructions. **(iv)** Comparisons against state-of-the-art language-to-action policies and LLM planners, and extensive ablations.

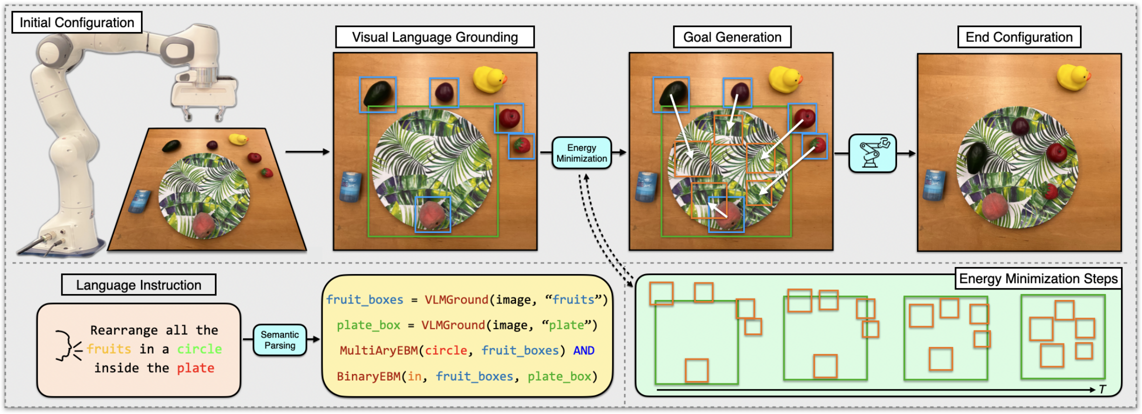
Simulation and real-world robot execution videos, as well as our code are publicly available on our website: `https://ebmplanner.github.io`.

3.2 Related Work

Following instructions for rearranging scenes: Language is a natural means of communicating goals and can easily describe compositions of actions and arrangements [3, 9, 10], providing more versatile goal descriptions compared to supplying one or more goal images. The latter requires the task to be executed beforehand, which defeats the purpose of instruction [73, 77, 89, 99]. We group methods in the literature in the following broad categories:

- *End-to-end language to action policies* [58, 64, 91, 93] map instructions to actions or to object locations directly. We have found that these reactive policies, despite impressively effective within the training distribution, typically do not generalize to longer instructions, new object classes and attributes or novel backgrounds [58, 91].
- *Symbolic planners* such as PDDL (Planning Domain Definition Language) planners [40, 65, 70, 94] use predefined symbolic rules and known dynamics models, and infer discrete task plans given an instruction with lookahead logic search [21, 40, 40, 65, 70, 94]. Symbolic planners assume that each state of the world, scene goal and intermediate subgoal can be sufficiently represented in a logical form, using language predicates that describe object spatial relations. These methods predominantly rely on manually-specified symbolic transition rules, planning domains and grounding, which limits their applicability.
- *Large language models (LLMs)* map instructions to language subgoals [32, 33, 102, 112] or program policies [51] with appropriate plan-like prompts. The predicted subgoals interface with low-level short-term policies or skill controllers. LLMs trained from Internet-scale text have shown impressive zero-shot reasoning capabilities for a variety of downstream language tasks [4] when prompted appropriately, without any weight fine-tuning [56, 97]. The scene description is usually provided in a symbolic form as a list of objects present, predicted by open-vocabulary detectors [41]. Recent works of [51, 52] have also fed as input overhead pixel coordinates of objects to inform the LLM's predictions. The prompts for these methods need to be engineered per family of tasks. It is yet to be shown how the composition of spatial concept functions can emerge in this way.

Language-conditioned scene generation: A large body of work has explored scene generation conditioned on text descriptions [39, 66, 81, 88, 106]. The work of [43] leverages web-scale pre-trained models [28, 80, 82] to generate segmentation masks for each object in the generated goal image. Given an input image, their method generates a text prompt using a captioning model and feeds it to a generative model



```
Rearrange all the
fruits in a circle
inside the plate
```

```
fruit_boxes = VLMGround(image, “fruits”)
plate_box = VLMGround(image, “plate”)
MultiAryEBM(circle, fruit_boxes) AND
BinaryEBM(in, fruit_boxes, plate_box)
```

Figure 3.2: **Scene rearrangement through energy minimization.** Given an image and a language instruction, a semantic parser maps the language into a set of energy functions (`BinaryEBM`, `MultiAryEBM`), one for each spatial predicate in the instruction, and calls to an open-vocabulary visual language grounder (`VLMGround`) to localize the object arguments of each energy function mentioned in the instruction, here “fruits” and “plate”. Gradient descent on the sum of energy functions with respect to object spatial coordinates generates the goal scene configuration. Vision-based neural policies condition on the predicted pick and place visual image crops and predict accurate pick and place locations to manipulate the objects.

that outputs a goal image, which is then further parsed into segmentation masks. However, the prompt is limited to contain only names of objects and there is no explicit language-guided spatial reasoning. In this work, we seek to make scene generation useful as goal imagination for robotic spatial reasoning and instruction following. Instead of generating pixel-accurate images, we generate object configurations by abstracting the appearance of object entities. We show this abstraction suffices for a great number of diverse scene rearrangement tasks.

Energy-based models: Our work builds upon existing work on energy-based models (EBMs) [14, 15, 16, 23, 55, 72]. Most similar to our work is that of [72], which generates and detects spatial concepts with EBMs on images with dots, and [15, 55], which demonstrates composability of image-centric EBMs for generating face images and images from CLEVR dataset [38]. In this work, we demonstrate zero-shot composability of EBMs over object poses instead of images, and showcase their applicability on spatial reasoning and instruction following for robotic scene rearrangement.

3.3 Method

The architecture of SREM is shown in Figure 3.2. The model takes as input an RGB-D image of the scene and a language instruction. A semantic parser maps the instruction to a set of spatial predicate energy functions and corresponding referential expressions for their object arguments. An open-vocabulary visual detector grounds the arguments of each energy function to actual objects in the scene. The goal object locations are predicted via gradient descent on the sum of energy functions. Lastly, short-term vision-based pick-and-place policies move the objects to their inferred goal locations. Below, we describe each component in detail.

A library of energy-based models for spatial concepts In our work, a spatial predicate is represented by an energy-based model (EBM) that takes as input x the set of objects that participate in the spatial predicate and maps them to a scalar energy value $E_\theta(x)$. An EBM defines a distribution over configurations x that satisfy its concept through the Boltzmann distribution $p_\theta(x) \propto e^{-E_\theta(x)}$. Low-energy configurations imply satisfaction of the language concept and have high probability. An example of the spatial concept can be generated by optimizing for a low-energy configuration through gradient descent on (part of) the input x. We represent each object entity by its 2D overhead centroid coordinates and box size. During gradient descent, we only update the center coordinates and leave box sizes fixed. We consider both binary spatial concepts (*in*, *left of*, *right of*, *in front of*, *behind*) as well as multi-ary spatial concepts (*circle*, *line*).

Using an EBM, we can sample configurations from p_θ, by starting from an initial configuration x^0 and refining it using Langevin Dynamics [98]:

$$x^{k+1} = x^k - \lambda \nabla_x E_\theta(x^k) + \epsilon^k z^k, \tag{3.1}$$

where z^k is random noise, λ is an update rate hyperparameter and ϵ^k is a time-dependent hyperparameter that monotonically decreases as k increases. The role of z^k and decreasing ϵ^k is to induce noise in optimization and promote exploration, similar to Simulated Annealing [46]. After K iterations, we obtain $x^- = x^K$. During training, we iterate over Equation 3.1 $K = 30$ times, using $\lambda = 1$ and $\epsilon_k = 5e - 3$. During inference, we find that iterating for more, e.g. $K = 50$ often leads to better solution. In this case we also linearly decay ϵ_k to 0 for $k > 30$.

We learn the parameters θ of our EBM using a contrastive divergence loss that penalizes energies of examples sampled by the model being lower than energies of ground-truth configuration:

$$\mathcal{L} = \mathbb{E}_{x^+ \sim p_D} E_\theta(x^+) - \mathbb{E}_{x^- \sim p_\theta} E_\theta(x^-), \tag{3.2}$$

where x^+ a sample from the data distribution p_D and x^- a sample drawn from the learned distribution p_θ. We additionally use the KL-loss and the L2 regularization proposed in [16] for stable training. At test time, compositions of concepts can be created by simply summing energies of individual constituent concept, as shown in Figures 3.1 and 3.2.

We implement two sets of EBMs, a BinaryEBM and a MultiAryEBM for binary (e.g., *left of*) and multi-ary (e.g., *circle*) language concepts, respectively. The BinaryEBM expects two object arguments, each represented by its bounding box. We convert the object bounding box to (top-left corner, bottom-right corner) representation. Then we compute the difference between all corners of the two object arguments and concatenate and feed to a multi-layer perceptron (MLP) that outputs a scalar energy value. Note that the energy function only depends on the relative arrangement of the two objects, not their absolute locations. The MultiAryEBM is used for order-invariant concepts of multiple entities, such as shapes. The input is a set of objects, each represented as a point (box center). We subtract the centroid of the configuration from each point and then featurize each object using an MLP. We feed this set of object features to a sequence of four attention layers [95] for contextualization. The refined features are averaged into an 1D vector which is then mapped to a scalar energy using an MLP. We train a separate EBM for each language concept in our vocabulary using corresponding annotated scenes in given demonstrations. Note that annotated scenes suffice to train the energy functions, kinesthetic demonstrations are not necessary, and in practice each EBM can be trained within a few minutes.

Semantic parsing of instructions into spatial concepts and their arguments. Our parser maps language instructions to instantiations of energy-based models and their arguments. It is a Sequence-to-Tree model [13] with a copying mechanism [24] which allows it to handle a larger vocabulary than the one seen during training. The input to the model is a natural language instruction and the output is a tree. Each tree node is an operation. The three operations supported are i) `BinaryEBM` which calls a BinaryEBM from our library, ii) `MultiAryEBM` and iii) `VLMGround` which calls the visual-language grounding module. Each node has a pointer to the arguments of the operation, language concepts for EBM calls, e.g., *behind*, and noun phrases for grounding model calls, e.g., *"the green cube"*. Nodes in the parsing tree may also have children nodes, which imply nested execution of the corresponding operations. The input utterance is encoded using a pre-trained RoBERTa encoder [60], giving a sequence of contextualized word embeddings and a global representation of the full utterance. Then, a decoder is iteratively employed to i) decode an operation, ii) condition on this operation to decode or copy the arguments for this operation, iii) add one (or more) children node(s). For example, the instruction *"a circle of cubes inside the plate"* is mapped to a sum of energy functions where each object of the multi-ary concept *circle* participates in the constraining binary concept *in*:

$$\begin{aligned} E^{total} = \; & \texttt{MultiAryEBM}(\textit{circle}, \texttt{VLMGround}(\text{“}\textit{cubes}\text{”})) \\ & + \textstyle\sum_i \texttt{BinaryEBM}(\textit{in}, x_i, \texttt{VLMGround}(\text{“}\textit{plate}\text{”})), \\ & x_i \in \texttt{VLMGround}(\text{“}\textit{cubes}\text{”}). \end{aligned} \tag{3.3}$$

We train our semantic parser on the instructions of all training demonstrations of all tasks jointly, as well as on synthesized instructions paired with programs, each with 1-7 predicates, that we generate by sampling from a grammar, similar to previous works [68, 96].

We ground noun phrases predicted by our parser with an off-the-shelf language grounding model [36], which operates as an open-vocabulary detector. The input is the noun phrase, e.g., *"the blue cube"* and the image, while the output is the boxes of all object instances that match the noun phrase. The open-vocabulary detector has been pre-trained for object detection and referential grounding on MS COCO [53], Flickr30k [76] and Visual Genome [47]. We finetune the publicly available code of [36] on our training data of all tasks jointly.

Short-term vision-based manipulation skills We use short-term manipulation policies built upon Transporter Networks [109] to move the obejcts to their predicted locations. Transporter Networks take as input one or more RGB-D images, reproject them to the overhead birds-eye-view, and predict two robot gripper poses: i) a pick pose and ii) a pick-conditioned placement pose. These networks can model any behaviour that can be effectively represented as two consecutive poses for the robot gripper, such as pushing, sweeping, rearranging ropes, folding, and so on – for more details please refer to [109].

We modify Transporter Networks to take as input a small image RGB-D patch, instead of a complete image view. Specifically, we consider as input the image patches around the object pick and object goal locations predicted by our visual grounding and energy-based minimization modules respectively. In this way, the low-level policies know roughly what to pick and where to place it, and only locally optimize over the best pick location, as well as the gripper's relative rotation, within an object of interest, or placement location, at a particular part of the scene, respectively. We show in our ablations (Table 3.7) that using learning-based pick-and-place policies helps performance, even if the search space is limited thanks to grounding and goal imagination. We train Transporter Networks from scratch on all our pick-and-place demonstration datasets jointly.

Termination of execution: SREM generates a goal scene by optimizing the relative poses of the objects mentioned in the instruction. We estimate how many objects should be moved by comparing the detected bounding box (by the language grounding model) and the optimized bounding box (by the EBM). For non-compositional tasks that involve binary concepts, we inject the prior that one object is fixed. Then we take as many actions as the number of objects the EBM moved.

Closed-loop execution: SREM first generates a goal scene from the input instruction and then executes it. After execution, we re-detect all relevant objects using our VLM-grounder module to check if they are close to their predicted goal locations. If the re-detected object's bounding box and initially predicted goal bounding box intersect over a certain IoU threshold, we consider the goal to be successfully executed. If we fail to reach the goal, we call again our vision based policies using the current scene configuration. Comparing the post-execution object configuration with the initially imagined goal scene allows to track progress and estimate goal completion as we show in the experimental section.

3.4 Experiments

We test SREM in its ability to follow language instructions for rearrangement of tabletop scenes in simulation and in the real world. We compare our model against LLM planners [33] and end-to-end language-to-action policies [91]. Our experiments aim to answer the following questions:

1. How does SREM compare to LLM planners in predicting scene configurations from instructions? (Section 3.4.1)
2. How does SREM compare to state-of-the-art language-to-action policies for rearranging scenes? How does their relative performance change with varying instruction length and varying amount of training data? (Section 3.4.2)
3. How does SREM generalize to novel objects, object colors and background colors, compared to an end-to-end language-to-action model? (Section 3.4.3)
4. How much do different modules of our framework contribute to performance? (Section 3.4.4)

Benchmarks: Existing language-conditioned manipulation benchmarks are usually dominated by a single spatial concept like "inside" [91]. To better illustrate the compositionality of spatial concepts, we introduce the following set of benchmarks, implemented with PyBullet:

- **spatial-relations**, containing single pick-and-place instructions with referential expressions in cluttered scenes with distractors, e.g. *"Put the cyan cube above the red cylinder"*. We consider the relations *left of*, *right of*, *in front of*, *behind*.
- **comp-one-step**, containing compositional instructions with referential expressions in cluttered scenes with distractors that require one object to be re-located to a particular location, e.g. *"put the red bowl to the right of the yellow cube, to the left of the red cylinder, and above blue cylinder"*.
- **comp-group**, containing compositional instructions with referential expressions in cluttered scenes with distractors that require multiple objects to be re-located, e.g., *"put the grey bowl above the brown cylinder, put the yellow cube to the right of the blue ring, and put the blue ring below the grey bowl"*.
- **shapes**, containing instructions for making multi-entity shapes (circles and

Human: Put the strawberry to the right of the apple and in front of the green bowl.
Scene: There is an apple, a green bowl and a strawberry in the scene.
Robot Thought: The goal state is ["strawberry right of apple", "strawberry in front of green bowl"]
Robot Action: Put the strawberry to the right of the apple.
Executor: Done.
Robot Action: Put the strawberry in front of the green bowl.
Executor: Done.

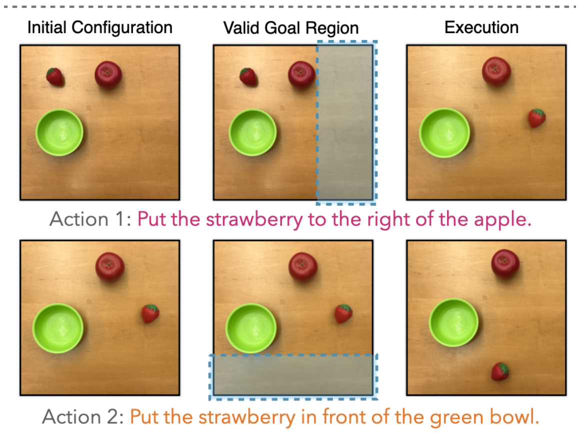

Initial Configuration
Valid Goal Region
Execution
Action 1: Put the strawberry to the right of the apple.
Action 2: Put the strawberry in front of the green bowl.

Figure 3.3: **Planning in language space with Large Language Models (LLMs).** LLM Planners predict language subgoals that decompose the initial instruction to simpler-to-execute subtasks. Predicted language subgoals are fed to reactive language-to-action policies for execution. In cases where concept intersection is needed, the predicted sequential language subgoal decomposition of instructions can fail. Here, the LLM predicts the first subgoal of putting the strawberry to the right of the apple. The reactive policy can succeed if it places the strawberry anywhere within the shaded region. During execution of the next issued language subgoal of putting the strawberry in front of the bowl, the policy violates the first constraint. Placing the strawberry in the intersection of the two shaded regions may not be achieved by decomposing the two predicates sequentially, as opposed to composing them. Then the burden of handling the compositional instruction is outsourced to the language-to-action policy, which often fails to generalize. Instead, SREM directly addresses compositionality of multiple spatial language predicates.

lines), e.g. *"rearrange all red cubes in a circle"*.

We further evaluate our model and baselines on four tasks from the CLIPort benchmark [91], namely **put-block-in-bowls**, **pack-google objects-seq**, **pack-google objects-group** and **assemble-kits-seq**.

For all tasks we train on either 10 or 100 demos and use the same demos to train all our modules, as discussed in Section 3.3. We test on 50 episodes per task, where we vary the instruction and the initial configuration of objects. For **spatial-relations** and **shapes** each concept corresponds to a task, while the composition benchmarks correspond to one task each.

Baselines: We compare SREM to the following baselines:

- CLIPort [91], a model that takes as input an overhead RGB-D image and an instruction and uses pre-trained CLIP language and image encoders to featurize the instruction and RGB image, respectively; then fuses these with depth features to predict pick-and-place actions using the action parametrization of Transporter Networks [109]. The model capitalizes on language-vision associations learnt by the CLIP encoders. We use the publicly available code of [91]. We train one CLIPort model on all tasks of each benchmark, e.g., one model for **spatial-relations**, a different for **comp-group** etc. Note that the original CLIPort implementation assumes access to oracle success/failure information based on which the model can retry the task for a fixed budget of steps or stop the execution if oracle confirms that the task is completed. We evaluate the CLIPort model without this oracle retry but still with oracle information of how many minimum steps it needs to take to complete the task, so we force CLIPort to take exactly that number of actions.
- LLMplanner, inspired by [33], an instruction-following scene-rearrangement model that uses an LLM to predict a sequence of subgoals in language form, e.g. *"pick the red cube and place it to the right of the blue bowl"*. The generated language subgoals are fed as input to language-to-action policies, such as CLIPort. Scene state description is provided as a list of objects in the scene. LLMplanner does not finetune the LLM but instead uses appropriate prompts so that the LLM adapts its behavior in-context and generates similar statements. The prompts include various previous successful interactions between a human user and the model. We design suitable prompts for our introduced benchmarks and use the LLM to decompose a long instruction into simpler ones (see Figure 2.3 in the Appendix for an example). Then, we feed each generated instruction to a CLIPort model, trained as described earlier. Lastly, for tabletop

	comp-one-step		**comp-group**	
Method	TP	TC	TP	TC
LLMplanner w/ oracle	82.0	59.0	75.3	29.0
SREM w/ oracle	**90.8**	**76.0**	**88.7**	**62.0**

Table 3.1: **Evaluation of SREM and LLMplanner with oracle perception and oracle low-level execution policies** on compositional spatial arrangement tasks. We report Task Progress (TP) and Task Completion (TC).

manipulation tasks in simulation, the LLMPlanner of [33] assumes access to an oracle success/failure detector. The difference in our implementation is that we do not assume any success detector. The execution terminates when all language subgoals have been fed to and handled by CLIPort.

Note that LLMplanner boils down to CLIPort for non-compositional instructions. As such, we compare with LLMplanner only on **comp-one-step** and **comp-group**, both in simulation and real world.

Evaluation Metrics: We use the following two evaluation metrics: (i) **Task Progress (TP)** [109] is the percentage of the referred objects placed in their goal location, e.g. $4/5 = 80.0\%$ for rearranging 4 out of 5 objects specified in the instruction. (ii) **Task Completion (TC)** rewards the model only if the full rearrangement is complete. For the introduced benchmarks we have oracle reward functions that evaluate whether the task constraints are satisfied.

3.4.1 Spatial reasoning for scene rearrangement with oracle perception and control

In this section, we compare spatial reasoning for predicting compositional scene subgoals in a language space versus in an abstract visually grounded space. In this section, to isolate this reasoning ability from nuisance factors of visually localizing the objects and picking them up effectively, we consider **oracle object detection, referential grounding and low-level pick-and-place policies.** Specifically, we carry out inferred language subgoals from LLMplanner using oracle controllers that relocate an object in the scene such that it satisfies the predicted subgoals. Note that SREM relies on pick-and-place policies that are not language-conditioned, while

	left-seen-colors		**left-unseen-colors**		**right-seen-colors**		**right-unseen-colors**	
Method	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos
CLIPort	13.0	44.0	9.0	33.0	29.0	43.0	28.0	44.0
SREM	**95.0**	**95.0**	**93.0**	**94.0**	**89.0**	**92.0**	**93.0**	**96.0**

	behind-seen-colors		**behind-unseen-colors**		**front-seen-colors**		**front-unseen-colors**	
Method	10	100	10	100	10	100	10	100
CLIPort	24.0	45.0	22.0	51.0	23.0	55.0	13.0	40.0
SREM	**87.0**	**87.0**	**89.0**	**90.0**	**89.0**	**90.0**	**88.0**	**89.0**

	circle-seen-colors		**circle-unseen-colors**		**line-seen-colors**		**line-unseen-colors**	
Method	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos
CLIPort	34.1	61.5	31.2	55.6	48.6	88.2	48.6	88.5
SREM	**91.3**	**91.5**	**90.2**	**91.2**	**98.1**	**99.0**	**98.4**	**99.4**

Table 3.2: **Evaluation (TP) of SREM and CLIPort on spatial-relations and shapes in simulation.**

	comp-one-step seen-colors		**comp-one-step unseen-colors**		**comp-group seen-colors**		**comp-group unseen-colors**	
Method	10	100	10	100	10	100	10	100
Initial (no movement)	0.0	0.0	0.0	0.0	31.7	31.7	31.8	31.8
CLIPort (zero-shot)	9.0	12.0	7.0	12.0	37.4	37.5	32.6	38.4
CLIPort	13.0	15.0	14.0	9.0	38.2	38.5	34.7	40.9
LLMplanner	51.2	53.2	49.4	53.5	38.6	39.0	37.1	39.0
SREM (zero-shot)	90.0	91.0	92.7	90.3	77.2	77.4	77.7	78.4
SREM (zero-shot + closed-loop)	**91.6**	**92.0**	**92.9**	**91.4**	**80.8**	**81.6**	**81.1**	**82.4**

Table 3.3: **Evaluation (TP) of SREM, CLIPort and LLMplanner on compositional tasks**. SREM is trained only on atomic relations and tested zero-shot on tasks with compositions of spatial relations which involve moving one (**comp-one-step**) or multiple (**comp-group**) objects to satisfy all constraints specified by the language. Some language constraints are satisfied already in the initial configuration and the Initial model captures that.

LLMplanner relies on language-conditioned policies for object re-location. Thus, the oracle control assumption is less realistic in the latter case. We forego this difference for the sake of comparison.

We show quantitative results of SREM and LLMplanner on the **comp-one-step** and **comp-group** benchmarks in Table 3.1. Our model outperforms LLMplanner and the performance gap is larger in more complex instructions. To elucidate why an abstract visual space may be preferable for planning, we visualize steps of energy

	put-block-in-bowl seen-colors		**put-block-in-bowl unseen-colors**		**packing-google-objects seq-seen-objects**		**packing-google-objects seq-unseen-objects**	
Method	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos	10 demos	100 demos
CLIPort	31.0	82.1	4.8	17.6	34.8	54.7	27.2	56.4
SREM	**84.3**	**93.8**	**89.0**	**95.3**	**86.8**	**94.8**	**88.0**	**92.9**

	packing-google-objects group-seen-objects		**packing-google-objects group-unseen-objects**		**assembling-kits seq-seen-colors**		**assembling-kits seq-unseen-colors**	
Method	10	100	10	100	10	100	10	100
CLIPort	33.5	61.2	32.2	70.0	38.0	**62.6**	36.8	**51.0**
SREM	**86.1**	**76.8**	**87.2**	**79.6**	**38.4**	42.0	**40.8**	44.0

Table 3.4: **Evaluation (TP) of SREM and CLIPort on CLIPort benchmark in simulation.**

minimization for different instructions in Figure 3.1 and steps of the execution of the LLM prompted by us to the best of our capability in Figure 3.3. We can see that SREM trained on single-predicate scenes shows remarkable composability in case of multiple predicates. Language planning on the other hand suffers from the ambiguity of translating geometric concepts to language and vice versa: step-by-step execution of language subgoals does not suffice for the composition of the two subgoals to emerge (Figure 3.3).

3.4.2 Spatial scene rearrangement

Simulation: In this section, we compare our model and the baselines in the task of instruction-guided scene rearrangement. We first show results on **spatial-relations** and **shapes** in Table 3.2. We largely outperform CLIPort, especially when less training demos are considered.

To evaluate generalization on longer instructions at test time, we show quantitative results in Table 3.3 for the benchmarks of **comp-one-step** and **comp-group**. We compare our model with CLIPort trained on atomic spatial relations and zero-shot evaluated on compositional benchmarks. We further fine-tune CLIPort on demos from the compositional benchmarks. SREM is not trained on these benchmarks, because the energy functions are already composable, meaning that we can jointly optimize over an arbitrary number of constraints by simply summing the different energy terms. Under all different settings, we significantly outperform all variants of CLIPort and LLMplanner. We also observe that closed-loop execution boosts our performance further.

We additionally show results on the CLIPort benchmark in Table 3.4. We largely outperform CLIPort on almost all tested tasks. Margins are significantly larger when i) less demos are used and ii) the robot has to interact with objects of unseen colors or classes. Most of the failure cases for our model are due to the language grounding mistakes - in particular for assemble-kits-seq we find that the grounder gets confused between letters and letter holes.

Real World: We test our model on a 7-DoF Franka Emika robot, equipped with a parallel jaw gripper and a top-down Azure Kinect RGB-D camera. We do not perform any real-world finetuning. Our test set contains 10 language-guided tabletop manipulation tasks per setting (Comp-one-step, Comp-group, Circles, Lines). We show quantitative results in Table 3.6. SREM generalizes to the real world without any real-world training or adaptation thanks to the open-vocabulary detector trained on real-world images, as well as the object abstractions in the predicate EBMs and low-level policy modules. We encourage readers to refer to our supplementary video and our website for more detailed results.

		spatial-relations		**composition**	
Novel attribute	Model	10 demos	100 demos	10 demos	100 demos
None	CLIPort	22.0	47.0	25.6	26.8
	SREM	90.0	91.0	83.6	84.2
Color	CLIPort	18.0	39.0	25.1	24.5
	SREM	87.0	85.0	86.5	84.0
Background	CLIPort	10.0	20.0	23.7	23.2
	SREM	79.0	68.0	77.0	72.0
Objects	CLIPort	17.0	19.0	24.5	24.8
	SREM	86.0	86.0	80.9	81.5

Table 3.5: **Generalization experiments of SREM and CLIPort in manipulation tasks in simulation (metric is TP).**

Method	**comp-one-step**	**comp-group**	**circles**	**lines**
CLIPort	13.1	22.9	34.0	46.0
LLMplanner	39.5	25.9	-	-
SREM	**85.6**	**75.8**	**94.0**	**90.0**

Table 3.6: **Real-world evaluation (TP) of SREM**

Method	Accuracy
SREM	77.2
SREM w/o goal generation	42.1
SREM w/o learnable policies	61.2
SREM w/ oracle language grounding	82.3
SREM w/ everything oracle except goal	88.3

Table 3.7: **Ablations of SREM on the benchmark comp-group-seen-colors (metric is TP).**

3.4.3 Generalization analysis

We conduct controlled studies of our model's generalization across three axes: a) **novel colors**: we train the models with objects of 7 different colors and evaluate them on objects of 4 unseen colors; b) **novel background colors**: we train all models on black-colored tables and evaluate on tables of randomly sampled RGB colors; c) **novel objects**: we train the models on objects of 4 classes and evaluate on rearrangement of 11 novel classes. In each of these settings, we only change one attribute (i.e. object color, background color or object instance) while keeping everything else constant.

We evaluate our model and CLIPort trained on 10 or 100 demos per task on **spatial-relations** (average performance over all tasks) and **composition** (average performance over all tasks from **comp-one-step** and **comp-group**). The results are summarized in Table-3.5. We observe that our model maintains high performance across all axes of generalization, independently of the number of training demos.

Our model's generalization capabilities rely on the open-vocabulary detector and the fact that EBMs and transporter-based low-level execution policy operate on abstracted space in a modular fashion. While CLIPort models can also generalize to novel scenarios by leveraging the CLIP model, the action prediction and perception are completely entangled and hence even if CLIP manages to identify the right objects based on the language, it has trouble predicting the correct pick and place locations.

3.4.4 Ablations

We show an error analysis of our model in Table-3.7. First, we remove the goal generation from SREM (SREM w/o goal generation) by conditioning the place network on the language input instead of the EBM-generated goal image, while keeping the pick network and object grounders identical. We observe a drop of 35.1% in accuracy, underscoring the importance of goal generation. We then remove our executor policy (SREM w/o learnable policies) and instead randomly select pick/place locations inside the bounding box of the relevant object. This results in a drop of 16%, showing the importance of robust low-level policies. We do not remove the grounder and parser since they are necessary for goal generation. We then experiment with oracle visual language grounder (SREM w/ oracle language grounding) that perfectly detects the objects mentioned in the sentence, which results in a performance gain of 5.1%. We finally evaluate with perfect grounding, language parsing and low-level execution (SREM w/ everything oracle except goal) to test the error rate of our goal generator. We obtain an 88.3% accuracy, thus concluding that our goal generator fails in 11.7% cases.

3.4.5 Limitations

Our model presently has the following two limitations: First, it predicts the goal object scene configuration but does not have any knowledge regarding temporal ordering constraints on object manipulation execution implied by physics. For example, our model can predict a stack of multiple objects on top of one another but cannot suggest which object needs to be moved first. One solution to this problem is to heuristically pick the order based on objects that are closer to the floor in the predicted scene configuration. However, more explicit encoding of physics priors are important to also identify if the generated configuration is stable or not. A promising direction is to model physics-based constraints as additional energy constraints, and obtain optimization gradients by leveraging either differentiable physics simulators [34, 79, 101] or learned dynamics models [50, 75, 100]. Second, our EBMs are currently parametrized by object locations and sizes, but different tasks need different abstractions. Manipulation of articulated objects, fluids, deformable objects or granular materials, would require finer-grained parametrization in both space and time. Furthermore, even for rigid objects, many tasks would require finer in-space parametrization, e.g., it would be useful to know a set of points in the perimeter of a plate as opposed to solely representing its bounding box for accurately placing things inside it. Considering EBMs over keypoint or object part graphs [67, 92] is a direct avenue for future work.

3.5 Conclusion

We introduce SREM, a modular robot learning framework for instruction-guided scene rearrangement that maps instructions to object scene configurations via compositional energy minimization over object spatial coordinates. We test our model in diverse tabletop manipulation tasks in simulation and in the real world. Our model outperforms state-of-the-art end-to-end language-to-action policies, and LLM-based instruction following methods both in in- and out-of-distribution settings, and across varying amount of supervision. We contribute a new scene rearrangement benchmark that contains more compositional language instructions than previous works, which we make publicly available to the community. Our work shows that a handful of visually-grounded examples suffice to learn energy-based spatial language concepts that can be composed to infer novel instructed scene arrangements, in long and complex compositional instructions.

Chapter 4

Conclusions

In this thesis, we propose a language-conditioned object detector that can detect all objects mentioned in a free-flowing natural language in 2D and 3D scenes. We show its application in tabletop manipulation setup where we combine the open-vocabulary object detector with a continously expandable library of concept learners that can compositionalize zero-shot and generalize in out-of-domain distribution settings. The key underlying idea with this thesis is to bring 2D and 3D perception closer so that 3D perception can benefit from strong and matured 2D architectures and internet scale 2D annotated data. Then, the strong vision systems can be seamlessly combined with concept learning and low-level manipulation to execute complex tasks.

In our ongoing and future work, we are actively considering the following problems:

- **Towards the Development of Large-scale Vision Foundation Models**: The current state-of-the-art in 3D computer vision heavily relies on domain-specific architectures and small-scale datasets, resulting in limited generalization capabilities. Conversely, 2D models trained on diverse datasets exhibit strong performance in real-world scenarios. Although our architecture, presented in Chapter 2, is applicable to both 2D and 3D language grounding with minimal modifications, we do not share parameters and data between the two modalities. This hinders the transfer of representations across modalities, which would be particularly advantageous for the low-resource 3D modality. Motivated by this, our ongoing work focuses on constructing a unified vision model using 2D techniques. Our model aims to enhance generalization, performance, and versatility by training on both 2D and 3D datasets. We aim to showcase experimental results on the Scannet benchmark [11], demonstrating superior performance compared to domain-specific architectures. Additionally, we aim to exhibit enhanced generalization capabilities by successfully adapting to new 3D scenes outside the training domain, surpassing the limitations of existing Scannet-based models. Lastly, we aim to demonstrate the ability to jointly train on both 2D and 3D datasets, achieving state-of-the-art results in both domains. Our objective is to highlight the potential of building unified vision foundation models that excel in multiple domains and enable seamless integration of 2D and 3D data. While the primary focus of this work is to demonstrate the benefits of 2D data for 3D perception, we are also interested in closing the loop by incorporating 3D understanding even when only 2D visual streams (without depth information) are available.

- **Learning Visual Perception for Unposed In-the-wild Dynamic Video Data**: In addition to the aforementioned work, which assumes access to posed RGBD images and static environments, there is a wealth of internet-scale video data that is dynamic and lacks available camera parameters. We aim to extend our Vision Foundation Model to learn generalizable representations by relaxing these assumptions, allowing for the effective analysis of unposed, in-the-wild dynamic video data.
- **Visual Imitation**: Finally, we plan to leverage and expand upon the visual models we develop for robotics applications, specifically focusing on learning a wide range of skills through visually imitating both active and passive human demonstrations (e.g., using YouTube videos). Our hypothesis is that achieving consistent 3D understanding will significantly contribute to this goal of learning from passive data.

Bibliography

[1] Ahmed Abdelreheem, Ujjwal Upadhyay, Ivan Skorokhodov, Rawan Al Yahya, Jun Chen, and Mohamed Elhoseiny. 3DRefTransformer: Fine-Grained Object Identification in Real-World Scenes Using Natural Language. In *Proc. WACV*, 2022. 2.1

[2] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. ReferIt3D: Neural Listeners for Fine-Grained 3D Object Identification in Real-World Scenes. In *Proc. ECCV*, 2020. 2.1, 2.2, 2.4.1, 2.1, 2.4.1

[3] Ahmed Akakzia, Cédric Colas, Pierre-Yves Oudeyer, Mohamed Chetouani, and Olivier Sigaud. Grounding Language to Autonomously-Acquired Skills via Goal Generation. In *ICLR 2021 - Ninth International Conference on Learning Representation*, Vienna / Virtual, Austria, May 2021. URL `https://hal.inria.fr/hal-03121146`. 3.2

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL `https://arxiv.org/abs/2005.14165`. 3.2

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *Proc. ECCV*, 2020. 2.1, 2.2, 2.3.1, 2.3.2

[6] Dave Zhenyu Chen, Angel Chang, and Matthias Nießner. ScanRefer: 3D Object Localization in RGB-D Scans using Natural Language. In *Proc. ECCV*, 2020. 2.1, 2.2, 2.4.1, 2.1

[7] Xinpeng Chen, Lin Ma, Jingyuan Chen, Zequn Jie, Wei Liu, and Jiebo Luo. Real-time referring expression comprehension by single-stage grounding network.

ArXiv, abs/1812.03426, 2018. 2.2

[8] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. UNITER: UNiversal Image-TExt Representation Learning. In *Proc. ECCV*, 2020. 2.2, 2.4.2, 2.3

[9] Cédric Colas, Tristan Karch, Nicolas Lair, Jean-Michel Dussoux, Clément Moulin-Frier, Peter Ford Dominey, and Pierre-Yves Oudeyer. Language as a cognitive tool to imagine goals in curiosity-driven exploration. *CoRR*, abs/2002.09253, 2020. URL https://arxiv.org/abs/2002.09253. 3.2

[10] Cédric Colas, Tristan Karch, Clément Moulin-Frier, and Pierre-Yves Oudeyer. Vygotskian autotelic artificial intelligence: Language and culture internalization for human-like ai, 2022. URL https://arxiv.org/abs/2206.01134. 3.2

[11] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *Proc. CVPR*, 2017. 2.3.2, 2.4.1, 4

[12] Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. In *Proc. ICCV*, 2021. 2.2

[13] Li Dong and Mirella Lapata. Language to logical form with neural attention. *arXiv preprint arXiv:1601.01280*, 2016. 3.3

[14] Yilun Du, Toru Lin, and Igor Mordatch. Model based planning with energy based models. *CoRR*, abs/1909.06878, 2019. URL http://arxiv.org/abs/1909.06878. 3.2

[15] Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6637–6647. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/49856ed476ad01fcff881d57e161d73f-Paper.pdf. 3.2

[16] Yilun Du, Shuang Li, Joshua B. Tenenbaum, and Igor Mordatch. Improved contrastive divergence training of energy based models. *CoRR*, abs/2012.01316, 2020. URL https://arxiv.org/abs/2012.01316. 3.1, 3.2, 3.3

[17] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From Captions to Visual Concepts and Back. In *Proc. CVPR*, 2015. 2.1, 2.2

[18] Mingtao Feng, Zhen Li, Qi Li, Liang Zhang, XiangDong Zhang, Guangming Zhu, Hui Zhang, Yaonan Wang, and Ajmal Mian. Free-form Description Guided

3D Visual Graph Network for Object Grounding in Point Cloud. In *Proc. ICCV*, 2021. 2.2, 2.1

[19] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. In *Proc. EMNLP*, 2016. 2.1, 2.2

[20] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-Scale Adversarial Training for Vision-and-Language Representation Learning. In *Proc. NeurIPS*, 2020. 2.4.2, 2.3

[21] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Stripstream: Integrating symbolic planners and blackbox samplers. *CoRR*, abs/1802.08705, 2018. URL `http://arxiv.org/abs/1802.08705`. 3.2

[22] Nikolaos Gkanatsios, Ayush Jain, Zhou Xian, Yunchu Zhang, Christopher Atkeson, and Katerina Fragkiadaki. Energy-based models as zero-shot planners for compositional scene rearrangement. *arXiv preprint arXiv:2304.14391*, 2023.

[23] Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. *CoRR*, abs/1912.03263, 2019. URL `http://arxiv.org/abs/1912.03263`. 3.2

[24] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016. 3.3

[25] Dailan He, Yusheng Zhao, Junyu Luo, Tianrui Hui, Shaofei Huang, Aixi Zhang, and Si Liu. TransRefer3D: Entity-and-Relation Aware Transformer for Fine-Grained 3D Visual Grounding. In *Proc. ACMMM*, 2021. 2.2, 2.1

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. CVPR*, 2016. 2.3.2

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 2.3.1

[28] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 3.2

[29] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *Proc. ICCV*, 2017. 2.2

[30] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. Modeling Relationships in Referential Expressions with Compositional Modular Networks. In *Proc. CVPR*, 2017. 2.1, 2.2

[31] Pin-Hao Huang, Han-Hung Lee, Hwann-Tzong Chen, and Tyng-Luh Liu. Text-Guided Graph Neural Networks for Referring 3D Instance Segmentation. In *Proc. AAAI*, 2021. 2.2, 2.1

[32] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022. 3.2

[33] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022. URL `https://arxiv.org/abs/2207.05608`. 3.1, 3.2, 3.4, 3.4

[34] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. *arXiv preprint arXiv:2104.03311*, 2021. 3.4.5

[35] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General Perception with Iterative Attention. In *Proc. ICML*, 2021. 2.3.2

[36] Ayush Jain, Nikolaos Gkanatsios, Ishita Mediratta, and Katerina Fragkiadaki. Bottom up top down detection transformers for language grounding in images and point clouds. In *European Conference on Computer Vision*, pages 417–433. Springer, 2022. , 3.1, 3.3

[37] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. In *Proc. CVPR*, 2016. 2.1, 2.2

[38] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017. 3.2

[39] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. *CoRR*, abs/1804.01622, 2018. URL `http://arxiv.org/abs/1804.01622`. 3.2

[40] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, pages 1470–1477, 2011. doi: 10.1109/ICRA.2011.5980391. 3.2

[41] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. MDETR - modulated detection for end-to-end multi-modal

understanding. *CoRR*, abs/2104.12763, 2021. URL https://arxiv.org/abs/2104.12763. 3.2

[42] Aishwarya Kamath, Mannat Singh, Yann André LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. MDETR - Modulated Detection for End-to-End Multi-Modal Understanding. In *Proc. ICCV*, 2021. 2.1, 2.2, 2.3, 2.3.2, 2.3.4, 2, 2.1, 2.2, 2.4.1, 2.3, 2.4, 2.5, 2.5

[43] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *ArXiv*, abs/2210.02438, 2022. 3.2

[44] Andrej Karpathy and Li Fei-Fei. Deep Visual-semantic Alignments for Generating Image Descriptions. In *Proc. CVPR*, 2015. 2.1, 2.2

[45] Sahar Kazemzadeh, Vicente Ordonez, Marc André Matten, and Tamara L. Berg. ReferItGame: Referring to Objects in Photographs of Natural Scenes. In *Proc. EMNLP*, 2014. 2.1, 2.2, 2.4.2

[46] Scott Kirkpatrick. Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics*, 34:975–986, 1984. 3.3

[47] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*, 123, 2016. 2.3.2, 2.4.2, 3.3

[48] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. VisualBERT: A Simple and Performant Baseline for Vision and Language. *ArXiv*, abs/1908.03557, 2019. 2.4.2, 2.4

[49] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded Language-Image Pre-training. In *Proc. CVPR*, 2022. 2.2

[50] Yunzhu Li, Jiajun Wu, Jun-Yan Zhu, Joshua B Tenenbaum, Antonio Torralba, and Russ Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019. 3.4.5

[51] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control, 2022. URL https://arxiv.org/abs/2209.07753. 3.2

[52] Bill Yuchen Lin, Chengsong Huang, Qian Liu, Wenda Gu, Sam Sommerer, and Xiang Ren. On grounded planning for embodied tasks with language models, 2022. URL https://arxiv.org/abs/2209.00465. 3.2

[53] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312. 3.3

[54] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Proc. ECCV*, 2014. 2.4.2

[55] Nan Liu, Shuang Li, Yilun Du, Joshua B. Tenenbaum, and Antonio Torralba. Learning to compose visual relations. *CoRR*, abs/2111.09297, 2021. URL https://arxiv.org/abs/2111.09297. 3.2

[56] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586, 2021. URL https://arxiv.org/abs/2107.13586. 3.2

[57] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In *Proc. ECCV*, 2016. 2.2

[58] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. *arXiv preprint arXiv:2110.10189*, 2021. 3.2

[59] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, abs/1907.11692, 2019. 2.3.1, 2.3.2

[60] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 3.3

[61] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-Free 3D Object Detection via Transformers. In *Proc. ICCV*, 2021. 2.1, 2.2, 2.3.2, 2.3.2, 1, 2.4.1

[62] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *Proc. NeurIPS*, 2019. 2.2, 2.3.3

[63] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-Task Vision and Language Representation Learning. In *Proc. CVPR*, 2020. 2.3.3

[64] Corey Lynch and Pierre Sermanet. Grounding language in play. *arXiv preprint*

arXiv:2005.07648, 2020. 3.2

[65] Daoming Lyu, Fangkai Yang, Bo Liu, and Steven Gustafson. SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning. *CoRR*, abs/1811.00090, 2018. URL http://arxiv.org/abs/1811.00090. 3.2

[66] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating images from captions with attention, 2015. URL https://arxiv.org/abs/1511.02793. 3.2

[67] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *Robotics Research: The 19th International Symposium ISRR*, pages 132–157. Springer, 2022. 3.4.5

[68] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019. 3.3

[69] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana-Maria Camburu, Alan Loddon Yuille, and Kevin P. Murphy. Generation and Comprehension of Unambiguous Object Descriptions. In *Proc. CVPR*, 2016. 2.4.2

[70] Toki Migimatsu and Jeannette Bohg. Object-centric task and motion planning in dynamic environments. *CoRR*, abs/1911.04679, 2019. URL http://arxiv.org/abs/1911.04679. 3.2

[71] Ishan Misra, Rohit Girdhar, and Armand Joulin. An End-to-End Transformer Model for 3D Object Detection. In *Proc. ICCV*, 2021. 2.2, 2.4.1

[72] Igor Mordatch. Concept learning with energy-based models. *CoRR*, abs/1811.02486, 2018. URL http://arxiv.org/abs/1811.02486. 3.2

[73] Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *CoRR*, abs/1807.04742, 2018. URL http://arxiv.org/abs/1807.04742. 3.2

[74] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. In *Proc. NIPS*, 2011. 2.4.2

[75] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2020. 3.4.5

[76] Bryan A. Plummer, Liwei Wang, Christopher M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models. In *Proc. ICCV*, 2015. 2.4.2, 3.3

[77] Vitchyr H. Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and

Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *CoRR*, abs/1903.03698, 2019. URL http://arxiv.org/abs/1903.03698. 3.2

[78] Charles Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. NIPS*, 2017. 2.3.2

[79] Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Scalable differentiable physics for learning and control. *arXiv preprint arXiv:2007.02168*, 2020. 3.4.5

[80] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020. 3.2

[81] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *CoRR*, abs/2102.12092, 2021. URL https://arxiv.org/abs/2102.12092. 3.2

[82] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022. 3.2

[83] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *Proc. CVPR*, 2016. 2.2, 2.2

[84] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. In *Proc. NIPS*, 2015. 2.1, 2.2, 2.3.2

[85] Seyed Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian D. Reid, and Silvio Savarese. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In *Proc. CVPR*, 2019. 2.3.1

[86] Junha Roh, Karthik Desingh, Ali Farhadi, and Dieter Fox. LanguageRefer: Spatial-Language Model for 3D Visual Grounding. In *Proc. CoRL*, 2021. 2.2, 2.1

[87] Arka Sadhu, Kan Chen, and Ram Nevatia. Zero-shot grounding of objects from natural language queries. In *Pro. ICCV*, 2019. 2.2

[88] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep

language understanding, 2022. URL https://arxiv.org/abs/2205.11487. 3.2

[89] Daniel Seita, Pete Florence, Jonathan Tompson, Erwin Coumans, Vikas Sindhwani, Ken Goldberg, and Andy Zeng. Learning to rearrange deformable cables, fabrics, and bags with goal-conditioned transporter networks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4568–4575. IEEE, 2021. 3.2

[90] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset for Automatic Image Captioning. In *Proc. ACL*, 2018. 2.4.2

[91] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021. 3.1, 3.2, 3.4, 3.4

[92] Maximilian Sieb, Zhou Xian, Audrey Huang, Oliver Kroemer, and Katerina Fragkiadaki. Graph-structured visual imitation. In *Conference on Robot Learning*, pages 979–989. PMLR, 2020. 3.4.5

[93] Elias Stengel-Eskin, Andrew Hundt, Zhuohong He, Aditya Murali, Nakul Gopalan, Matthew Gombolay, and Gregory Hager. Guiding multi-step rearrangement tasks with natural language instructions. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1486–1501. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/stengel-eskin22a.html. 3.2

[94] Marc Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1930–1936. AAAI Press, 2015. ISBN 9781577357384. 3.2

[95] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3.3

[96] Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1332–1342, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1129. URL https://aclanthology.org/P15-1129. 3.3

[97] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc

Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022. URL https://arxiv.org/abs/2201.11903. 3.2

[98] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011. 3.3

[99] Hongtao Wu, Jikai Ye, Xin Meng, Chris Paxton, and Gregory Chirikjian. Transporters with visual foresight for solving unseen rearrangement tasks, 2022. URL https://arxiv.org/abs/2202.10765. 3.2

[100] Zhou Xian, Shamit Lal, Hsiao-Yu Tung, Emmanouil Antonios Platanios, and Katerina Fragkiadaki. Hyperdynamics: Meta-learning object and agent dynamics with hypernetworks. *arXiv preprint arXiv:2103.09439*, 2021. 3.4.5

[101] Zhou Xian, Bo Zhu, Zhenjia Xu, Hsiao-Yu Tung, Antonio Torralba, Katerina Fragkiadaki, and Chuang Gan. Fluidlab: A differentiable environment for benchmarking complex fluid manipulation. In *International Conference on Learning Representations*, 2023. 3.4.5

[102] Danfei Xu, Roberto Martín-Martín, De-An Huang, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Regression planning networks. *CoRR*, abs/1909.13072, 2019. URL http://arxiv.org/abs/1909.13072. 3.2

[103] Zhengyuan Yang, Boqing Gong, Liwei Wang, Wenbing Huang, Dong Yu, and Jiebo Luo. A fast and accurate one-stage approach to visual grounding. In *Proc. ICCV*, 2019. 2.2

[104] Zhengyuan Yang, Tianlang Chen, Liwei Wang, and Jiebo Luo. Improving one-stage visual grounding by recursive sub-query construction. In *Proc. ECCV*, 2020. 2.2

[105] Zhengyuan Yang, Songyang Zhang, Liwei Wang, and Jiebo Luo. SAT: 2D Semantics Assisted Training for 3D Visual Grounding. In *Proc. ICCV*, 2021. 2.2, 2.2, 2.1, 2.4.1

[106] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation, 2022. URL https://arxiv.org/abs/2206.10789. 3.2

[107] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling Context in Referring Expressions. In *Proc. ECCV*, 2016. 2.1, 2.4.2

[108] Zhihao Yuan, Xu Yan, Yinghong Liao, Ruimao Zhang, Zhen Li, and Shuguang

Cui. InstanceRefer: Cooperative Holistic Understanding for Visual Grounding on Point Clouds through Instance Multi-level Contextual Referring. In *Proc. ICCV*, 2021. 2.2, 2.1

[109] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. *arXiv preprint arXiv:2010.14406*, 2020. 3.1, 3.3, 3.4

[110] Lichen Zhao, Daigang Cai, Lu Sheng, and Dong Xu. 3DVG-Transformer: Relation Modeling for Visual Grounding on Point Clouds. In *Proc. ICCV*, 2021. (document), 2.2, 2.1

[111] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *Proc. ICLR*, 2021. 2.2, 2.3.2, 2.3.2, 2.3.2, 2.4.2

[112] Yifeng Zhu, Jonathan Tremblay, Stan Birchfield, and Yuke Zhu. Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs. *CoRR*, abs/2012.07277, 2020. URL `https://arxiv.org/abs/2012.07277`. 3.2