Task-specific Sensor Planning for Robotic Assembly Tasks

Guy Rosman[1] and Changhyun Choi[1] and Mehmet Dogar[2] and John W. Fisher III[1] and Daniela Rus[1]

Abstract— **When performing multi-robot tasks, sensory feedback is crucial in reducing uncertainty for correct execution. Yet the utilization of sensors should be planned as an integral part of the task planning, taken into account several factors such as the tolerance of different inferred properties of the scene and interaction with different agents.**

In this paper we handle this complex problem in a principled, yet efficient way. We use surrogate predictors based on open-loop simulation to estimate and bound the probability of success for specific tasks. We reason about such task-specific uncertainty approximants and their effectiveness. We show how they can be incorporated into a multi-robot planner, and demonstrate results with a team of robots performing assembly tasks.

I. Introduction

For mobile manipulation in multi-robot tasks such as assembly operations (e.g. inserting fasteners), fixed sensors are often insufficient, due to occlusions, and resolution limitations. Planning algorithms can exploit mobility to ensure that the sensor is positioned in a way that is useful according to the current task. Such algorithms must contend with changes of critical positions and aspects of interest as a function of sub-tasks.

For example, inserting a part with a protrusion into a hole is more compliant in some directions than others, requiring specific point of views in order to predict the successful termination of the operation. Sensor planning has to further take into account the parts and other robots' positions in order to avoid occlusion of the sensor and collisions between robots and parts during plan execution. We approach this challenging multi-robot complex planning problems using an estimation approach, by reasoning about a virtual sensor and how it would reduce uncertainty of the assembly operation, as part of the overall multi-robot planner.

We present an example scenario in Figure 1. In this setup, multiple assembly robots plan to grasp the parts (a peg and a block with a hole) and bring them to assembly configurations, such as the one seen in the figure. The position of the sensor and the configuration of the robot carrying it is also planned. The sensor is used at this step to estimate the poses of the parts. Then, according to the estimates, the robots that are carrying the parts correct their configurations to better align the parts with each other. After this alignment, a local controller moves the robots on a straight line to mate the parts and uses force-feedback to stop this motion. Our goal is to plan the assembly configuration for both robots and sensors. The problem of manipulation planning and sensor positioning is intertwined: a good sensor position to view the parts may not leave any room for the assembly robots to grasp parts, and good grasping poses for robots may occlude the view of the sensor. Therefore, we present a planner which simultaneously solves the manipulation and sensor positioning problem. Within the planner, each sensor position is evaluated using the fast task-specific metrics described above.

While planning and state estimation are often approached hierarchically, sensor planning was so far limited to aid low-level tasks — such as estimating geometry, and localizing the robot. We aim to narrow this gap by performing 3D sensor planning for elementary tasks. We then utilize the resulting estimator within a manipulation planner for assembly tasks. By elementary tasks we refer to tasks with few motions, that are stable enough to be controlled in an open-loop manner – e.g. placing fasteners, pushing buttons, pulling levers, to name a few. For such tasks we can reason about sensor positions in terms of the task at hand, given initial uncertainty, the scene structure, and other factors.

In order to plan sensors' positions, we must estimate task success and uncertainty as a function of sensor position. We employ a surrogate open-loop task success function, with the same measurements model used in robot and object pose estimation from range images. This allows us to plan for 3D sensor poses as part of an assembly planner. We show that under certain conditions our estimate is a lower bound on the probability of success of the real operation, thereby providing guarantee for planning purposes.

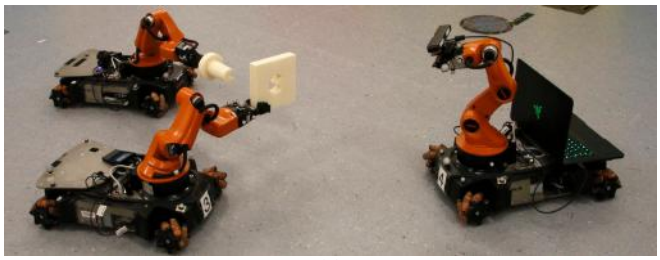
While we focus on assembly tasks as an example, other manipulation tasks could easily be accommodated, such as pressing buttons, or closing valves. We look at short-horizon elementary actions, and leave persistent operation, including online state estimation to future research.

The main contributions of the paper are as follows,

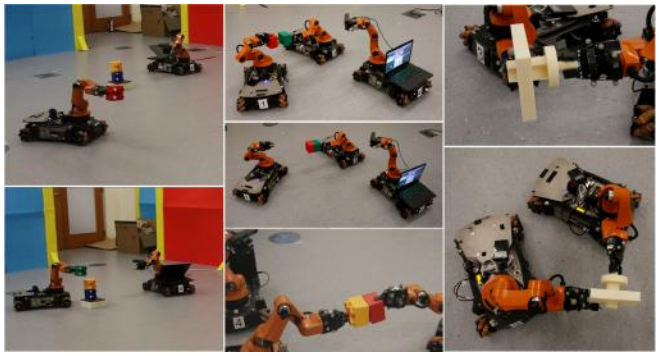
1 A model that enables the use of surrogate functions for planning to position a mobile 3D sensor for specific robotic tasks, with a well-defined probabilistic framework. We show conditions under which these surrogate functions bound the task success.
2 An algorithm estimating the utility of sensor configurations in combination with robotic manipulators.
3 An end-to-end system that uses mobile sensors to assist with fixture-less assembly tasks. We show how we simultaneously perform manipulation planning and sensor positioning for fixture-less assembly tasks.

[1]Computer Science and Artificial Intelligence Lab, MIT, Cambridge, MA 02139, USA rosman,cchoi,fisher,rus@csail,mit.edu

[2]Faculty of Engineering, Leeds University, Stoner Building, Leeds, UK M.R.Dogar@leeds.ac.uk



(a)



(b)

Fig. 1: (a) A team of robots assembling a peg into a hole. The rightmost robot carries a sensor to localize the assembly parts and provides feedback. The grasps, robot base/arm configurations, and the sensor position are all planned to avoid collisions *and* to position the sensor optimally for the task. (b) Example tasks - block stacking, block assembly, peg in hole.

In Section III we describe our model of the sensing process, and the probabilistic modeling of the objects and sensor, including uncertainty estimation for various quantities. Given a scene such as the one in Fig. 1, this model allows us to quantify how useful the sensor pose is for the specific task. In Section IV we reason about how to incorporate uncertainty estimation into a high-level robotic planner, so as to choose configurations for all the robots which are both feasible and informative. In Section V-A we describe a model system for multi-robot tasks, used in our experiments, based on the integration of uncertainty estimation into the high-level assembly planner. In Section V-B we demonstrate in several experiments how uncertainty estimation from our model allows us to compute meaningful and efficient plans for our robots.

II. Related work

There is an extensive literature on reasoning about sensor placement. In computer vision this is often addressed as part of active perception [3], [5], [34]. In robotics, such efforts are part of next-best-view planning and dynamic planning [1], [15], [41], [24], [33], [19], although in our case we estimate planner-level notions (task success) rather than lower-level ones (geometric or classification uncertainty). Estimating the uncertainty of an event given a set of observations has been a key topic in information-theoretic research [27], [9], with uses in various fields. Especially a greedy sensor selection methods such as ours has been shown to approximate the optimal approaches for sensor planning [43], within some factor. Finally, sampling over physical simulations has been utilized for applications such as grasp planning [23] and agent-based object recognition [6].

In these efforts however, there is still a gap between task-planning and sensor planning. Sensor planning should model the reduction in uncertainty related to task success based on the same models used to plan the tasks. It is this gap which our approach explores.

As our work reasons about possible actions in a continuous information space under uncertainty, it relates to *partially observable Markov decision processes* (POMDPs) and ways to approximate them (see for example [28], [31], [35], [37]), under the right constraints in configuration space and time, and cost function choice. However, our approach is more tailored to the nature of problems common to robotic assembly and other manipulation tasks. Here the state is continuous, uncertainty is captured well in terms of the geometry, and sensing from the right viewpoint allows us to capture most of the uncertainty in task success. This gives us a different solution, with a bound that differs from the related QMDP approach, and is described in the Section III.

In terms of the sensors and perception subsystems, traditionally a monocular camera with fiducials or 3D object models have often been used to establish object poses. In recent years, the spread of RGB-D sensors has made them reliable and robust sources for many applications in robotics such as object localization [2], [13], object recognition [25], and Simultaneous Localization and Mapping (SLAM) [26], [30], [20]. Iterative Closest Point (ICP) algorithms usually assume some knowledge of an initial location. Starting from a coarse initial estimate, the ICP algorithm minimizes some distance criterion such as point-to-point [10], point-to-plane [12], or more general ones [29].

Our work is also related to grasp planners that take into account task constraints [7], [8], [17] and multiple robots [40]. Particularly, the planner we present in Section IV is an extension of the planner from [18]. We build on this literature and extend it to plan simultaneously for multi-robot constraints *and* sensor positioning. This enables our multi-robot system to take into account uncertainty during manipulation, a topic which is also addressed in [11], [22], [14].

III. Posterior Uncertainty Estimation

We now describe how to evaluate sensor poses. Our main goal is to find sensor poses that will provide range images to guide fine manipulation tasks as in assembly operations. We assume global robot positions are usually available. However, we need to sense the fine-grained poses of the objects and manipulators that are crucial for successful manipulation, and this is where range sensors have proven very useful in recent years. We wish to quantify the utility of possible sensor poses so that we can place the sensors as part of the assembly plan.

Let us first describe task-agnostic measures of uncertainty. We then describe reasoning about task-specific uncertainty measures obtained by sampling possible control trajectories, their likelihood given observations, and their success probability. These measures are used by the high-level planner

as described in Section IV to decide which sensor pose is preferred, along with the configurations of the other robots, picking configurations which are both feasible and informative. While we mention a prior uncertainty, uncertainty estimates are usually the result of state updates and prior knowledge. In our case we assume they are given.

We mark a configuration of the system as $\mathbf{x}$, e.g. the pose of the robots and objects they interact with and observe. The space of all possible configurations is known as the configuration space $\mathcal{X}$ for the system. $\mathbf{o}$ denotes our observations of the scene, i.e. range images, and possibly other measurements, e.g. localization systems such as GPS or VICON.

In order to reason about the range sensor viewpoints we employ a likelihood term from the ICP [29] and SLAM literature [39] — we assume the range observations $\mathbf{o} = \{z_i\}$ are Gaussian i.i.d. given the state $\mathbf{x}$, with mean given by the transformed model points in the scene.

$$p(\mathbf{o}|\mathbf{x}) \propto \prod_i \exp\left\{-d^2(z_i, T_{\mathbf{x}}(S_{model}))\right\}, \tag{1}$$

where S_{model} denotes the objects in the scene, for which we assume a model (e.g. a mesh representation), as well as the robots' bodies. i denotes the pixel index in the image. $T_{\mathbf{x}}$ denotes the transformation of each object according to the pose values. In our case, the state of the system $\mathbf{x}$ is defined as the poses of all objects and the sensor, typically $SE(3)^{N+1}$ where N is the number of objects. $d^2(\cdot,\cdot)$ is the quadratic depth difference, often replaced by robust distance functions in the ICP and SLAM literature.

Using this observations model, we can estimate several quantities—such as posterior uncertainty in the poses after observing the objects and its effect on the probability of success of various manipulation tasks. We differentiate between several configurations of interest. We denote by $\mathbf{x}_p$ the (determininistic) planned configuration, given by the planner at the beginning of some step. We denote by $\mathbf{x}_T$ the true configuration after the last motion step. $\mathbf{x}_S$ is the sensed state, estimated given sensor observations, representing our posterior belief of the robot and scene. We assume $\mathbf{x}_S$ and $\mathbf{x}_T$ are distributed around $\mathbf{x}_p$, and $p(\mathbf{X})$ denotes the distribution of the true state at a certain moment. Let $f(\mathbf{x}_S, \mathbf{x}_T)$ denote any discrepancy function between $\mathbf{x}_S$ and $\mathbf{x}_T$. f describes the cost associated with assuming state $\mathbf{x}_S$ while state $\mathbf{x}_T$ is the true state. It can include error terms of $\mathbf{x}_S$ and $\mathbf{x}_T$, or of $\mathbf{x}_S$ and the observations given $\mathbf{x}_T$, $\mathbf{o}(\mathbf{x}_T)$. $\mathrm{E}(f)$ is a function of the sensor location in $\mathbf{x}_p$, and our goal is to optimize it by choosing a suitable $\mathbf{x}_p$. We can obtain the expectation of f by total expectation

$$\mathrm{E}(f) = \sum_{\substack{\mathbf{x}_T \sim p(\mathbf{X}) \\ \mathbf{x}_S \sim p(\mathbf{X}|\mathbf{o}(\mathbf{x}_T))}} f(\mathbf{x}_S, \mathbf{x}_T). \tag{2}$$

Suppose that our planner attempts to place the sensor at location $\mathbf{x}_p$. $\mathbf{o}(\mathbf{x}_T)$ is the set of observations seen given state $\mathbf{x}_T$, and we assume it to be a deterministic function. Samples of $\mathbf{x}_S$, for a specific $\mathbf{x}_T$, are illustrated in Figure 2(b).

We now go over several choices of f, and show their merit in the context of task planning.

A. Pose Uncertainty

Several forms of conditional uncertainty estimates are captured by Equation 2. As a specific example common to sensory planning, plugging in $f(\mathbf{x}_S, \mathbf{x}_T) = -\log(p(\mathbf{x}_S|\mathbf{o}(\mathbf{x}_T)))$ we get the conditional entropy of the pose given measurements

$$H(\mathbf{X}|\mathbf{O}) = \sum_{\substack{\mathbf{x}_T \sim p(\mathbf{X}) \\ \mathbf{x}_S \sim p(\mathbf{X}|\mathbf{o}(\mathbf{x}_T))}} -\log p\left(\mathbf{x}_S|\mathbf{o}(\mathbf{x}_T)\right), \tag{3}$$

which can be computed by sampling different $\mathbf{x}_T$ values and their observations $o(\mathbf{x}_T)$, and integrating the conditional entropy. Here capital letters such as $\mathbf{X}, \mathbf{O}$ emphasize the random variable, as opposed to sampled values of that variable. Entropy estimation for non-parametric distributions has been studied intensely (see [21] [38], and references therein). In our case, for numerical efficiency and stability we chose a parametric approximation by a Gaussian in the Lie algebra $se(3)$ of each object, assuming objects to be statistically independent. The entropy of a multivariate Gaussian distribution is given by [16]

$$H(\mathbf{x}_S|\mathbf{o}) = \frac{d_1}{2}\left(1 + \log\left(2\pi\right)\right) + \frac{1}{2}\log\left(|\mathbf{\Sigma}_S|\right), \tag{4}$$

where $\mathbf{\Sigma}_S$ is the empirical estimate of the variance of $\mathbf{x}_S$, and d_1 is the dimensionality. Equation 2 therefore becomes

$$\mathrm{E}_{\mathbf{x}_T}\left\{\frac{d_1}{2}\left(1 + \log\left(2\pi\right)\right) + \frac{1}{2}\log\left(|\mathbf{\Sigma}_S(\mathbf{x}_T)|\right)\right\}. \tag{5}$$

We obtain our samples of $p(\mathbf{X}|\mathbf{o})$ by sampling from $p(\mathbf{X})$, using Bayes' rule, and using $p(\mathbf{o}|\mathbf{x}_S)$ to reweight the samples of $\mathbf{x}_S$, up to a normalization constant. While the number of samples required to depict distributions in $6N$ degrees of freedom (DOF) is prohibitively high, in our case, assuming i.i.d. distribution of the objects was sufficient in terms of accuracy and efficiency.

Other measures that would be interesting to estimate are the squared Frobenius error between $\mathbf{x}_S$ and $\mathbf{x}_T$ as $f(\mathbf{x}_S, \mathbf{x}_T) = \|\mathbf{x}_T - \mathbf{x}_S\|_F^2$, and the mutual information between $\mathbf{o}$ and $\mathbf{x}_S$ as $f(\mathbf{x}_S, \mathbf{x}_T) = \log p(\mathbf{x}_S) - \log p(\mathbf{x}_S|\mathbf{o}(\mathbf{x}_T))$.

B. Task-Specific Estimator

Perfect pose information is often not required for a specific task. Hence, uncertainty measures such as entropy of the full state space may not be the right criteria to plan for. Consider for example inserting a key in a hole — the angle of rotation around the cylinder axis (see illustration in Figure 2) is crucial for determining whether the key would go in, but excess translation along the cylinder axis is less crucial as it will be countered by the lock. In fact, from some viewpoints (behind the key), some DOF may be barely observable, even though they are important to the task. This motivates assessing viewpoints according to a task-specific measure.

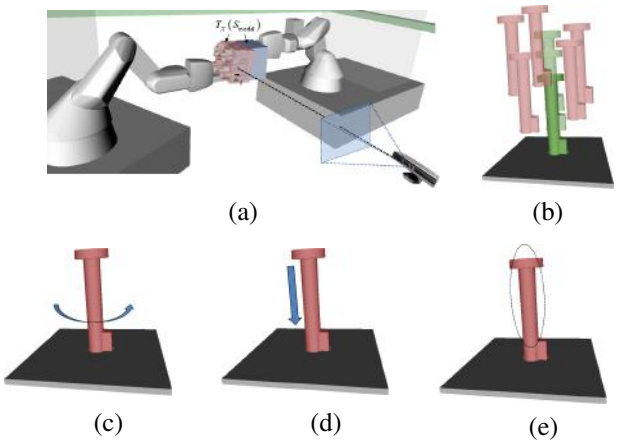


Fig. 2: **a)** An illustration of the likelihood model. $T_{\mathbf{x}}(S_{model})$ describes the state of scene objects, captured by the red and blue cubes. z_i defines the point at range image location i, captured by the green dot. **b)** A set of sampled poses for key insertion (translation uncertainty only). Some poses (green) would allow successful insertion with an open-loop controller, unlike other poses (red). **c)** Inserting a key into a keyhole is sensitive to the rotation. **d)** Translation along the axis of approach is less critical as excess force is usually countered by the keyhole. e) A possible success region for inserting a key into a hole, with varying tolerances in different DOF.

We would like to maximize task success directly, by choosing as f the probability of task failure having placed the sensor at position $\mathbf{x}_p$ and minimizing $\mathrm{E}(f)$.

Consider an elementary robotic task, such as assembling two parts. We denote the success of that task by $S(\mathbf{x}_S, \mathbf{x}_T, u)$. It is a function of the system state $\mathbf{x}_T$ and the control signal u chosen to get us to the planned state. The control signal u may depend on the believed system state $\mathbf{x}_S$ and some strategy for planning. We will use the control signal interchangeably with the planner and controller that creates it. In order to reason about S, we have to consider expected future observations and their effect on the control signal, given that in practice, the initial state was actually $\mathbf{x}_T$.

We claim that in many intermediate-level planning subtasks of assembly, taking an open-loop approach, combined with a plausible surrogate planner, can efficiently estimate informative sensor positions for the task. For example, for the task of insertion, we propose the following approximation: given two objects, A and B to be assembled, we take the following, open-loop plan (see Figure 3): 1) Compute a direction of insertion, v_{AB}. 2) Align objects so that object A is at a safe distance along v_{AB} from object B, and both of the objects are aligned. 3) Move object A towards object B in direction v_{AB}. The surrogate planner described in Algorithm 1, and illustrated in Figure 3.

Since the proposed strategy is open-loop, we can estimate its success by planning for believed state $\mathbf{x}_S$, but applying it to initial state $\mathbf{x}_T$. This approximation acts as a surrogate to the real control law in use by the robots. The collisions incurred by the execution and their location allow us to approximate whether a task is likely to succeed, and form $\hat{S}$, an *approximate success criteria* which we will now use. While the checks for collisions are similar to guarded moves

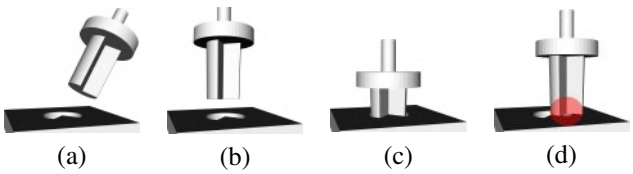


Fig. 3: An open-loop plan for part insertion. **a)** Initial (believed) position $\mathbf{x}_S$. **b)** Initial alignment. **c)** After insertion. **d)** Performing the open-loop plan for $\mathbf{x}_S$ whereas the real initial pose is $\mathbf{x}_T$ could result in a collision (shown in red), or task failure.

[42], we also look at the colliding surfaces, to discount collisions at small angles, where surfaces are likely to slide off one another.

As long as the simulation and criteria of success are accurate enough, the approach we take computes a lower bound on the probability of success, shown as Lemma 1. The assumptions we make in the lemma are fulfilled in practice for many relatively short horizon actions in assembly and other fine manipulation tasks. Having observed the world modifies the (posterior) distribution of $\mathbf{x}_S$. The chance of success for a given sensor location is therefore given by

$$\underset{\mathbf{x}_S,\mathbf{x}_T}{\mathrm{E}} \left(p(S|\mathbf{x}_S, \mathbf{o}(\mathbf{x}_T))\right) \approx \sum_{\substack{\mathbf{x}_T \sim p(\mathbf{x}) \\ \mathbf{x}_S \sim p(\mathbf{x}|O(\mathbf{x}_T))}} \hat{S}(\mathbf{x}_S, \mathbf{x}_T, \hat{u}) \qquad (6)$$

where $p(\mathbf{X}|\mathbf{O})$ can be sampled by importance weights using $p(\mathbf{X})$. The sampling order is defined as Algorithm 2, and provides us with a cost for a specific sensor pose, to be used by the planner in Section IV. f is computed via our approximation for task success $\hat{S}(\mathbf{x}_S, \mathbf{x}_T, \hat{u})$, described in terms of planning insertion/attachment of object A to object B, as Algorithm 1. We assume the definition of the assembly task includes a known approach direction v_{AB} for the insertion procedure.

Algorithm 1 Estimating success of inserting object A into object B, with approach direction v_{AB}

1: Let waypoint $w_1 = (R_1, t_1)$ mark the initial state of A
2: Let waypoint $w_3 = (R_3, t_3)$ mark the final state of A
3: Construct waypoint w_2 by rotation R_3 and a translation component created by projecting $t_1 - t_3$ onto v_{AB}
4: Simulate the trajectory of the object by interpolating between (w_1, w_2, w_3).
5: If any obstructive collisions are encountered, return failure, $S = 0$.
6: Otherwise, return $S = 1$

Lower Bound Interpretation Under certain conditions, Algorithm 1 provides a lower bound for the task success of the real planner and robot, beyond the bounds that are available for approximate POMDP solvers such as QMDP [35]. To see this, we re-examine the proposed approach. We denote by $u(\mathbf{x}_S)$ the actual planner of our robot, and $u_{opt}(\mathbf{x}_S)$ denotes a the closed-loop optimal plan starting at state $\mathbf{x}_S$. We compute an approximate open-loop *surrogate*

Algorithm 2 Estimating uncertainty for a sensor pose

for $i = 1, 2, \ldots, N_T$ **do**
  Sample state $\mathbf{x}_T$ from $p(\mathbf{x})$
  for $i = 1, 2, \ldots, N_S$ **do**
    Sample state $\mathbf{x}_S$ from $p(\mathbf{x})$, estimate the partition function / integration constant.
  for $i = 1, 2, \ldots, N_S$ **do**
    Sample state $\mathbf{x}_S$ from $p(\mathbf{x})$,
    Estimate $p(\mathbf{O}|\mathbf{x}_S)$, aggregate an estimate for $p(\mathbf{x}|\mathbf{O})$ (Subsec. III-A), or
    Estimate $f(\mathbf{x}_S, \mathbf{x}_T)$, aggregate $\mathrm{E}_{\mathbf{x}_S,\mathbf{O}}\, f$ (Subsec. III-B)
  aggregate $\mathrm{E}_{\mathbf{x}_T}\, \mathrm{E}_{\mathbf{x}_S,\mathbf{O}}\, f$
Output $\mathrm{E}_{\mathbf{x}_T}\, \mathrm{E}_{\mathbf{x}_S,\mathbf{O}}\, f$

strategy $\hat{u}$ for a given start state $\mathbf{x}_S$, such as the one shown in Algorithm 1. Given the strategy we compute an approximate success criteria $\hat{S}(\mathbf{x}_S, \mathbf{x}_T, \hat{u})$ by simulating the system, checking for errors and collisions. Marginalizing this term over all true states $\mathbf{x}_T$ and believed states $\mathbf{x}_S$ gives us a lower-bound estimate of the sensor location utility.

We make the following assumptions:

1) The robots' planner provides plans u close enough to an optimal closed-loop planner u_{opt}. Due to the tasks we explore and static scene except for our robots, sufficient knowledge of the scene affords a successful plan. Our planner can provide such a plan, and the objects can be localized using ICP from our 3D sensor to a reasonable, yet not always sufficient, degree of accuracy.
2) The environment changes slowly enough so that the optimal closed-loop u_{opt} planning is better in expectation than the optimal open-loop planning. Specifically it is better in expectation than other open-loop planners such as $\hat{u}$.
3) The success estimator $\hat{S}(\mathbf{x}_S, \mathbf{x}_T, \cdot)$ is accurate enough in predicting the success of a strategy in the real world under initial state uncertainty.
4) The surrogate planner $\hat{u}(\mathbf{x}_S)$ is open loop.

Assuming an open-loop control model and sub-optimal planner provides the following lower bound on planner success probability.

Lemma 1: If conditions (1)-(3) above hold, the surrogate estimator in Algorithm 2 provides a lower bound for the success rate of an the system planner.

Proof:

$$\mathrm{E}_{\mathbf{x}_S,\mathbf{x}_T}\, \hat{S}(\mathbf{x}_S, \mathbf{x}_T, \hat{u}) \overset{(3)}{\approx} \mathrm{E}_{\mathbf{x}_S,\mathbf{x}_T}\, S(\mathbf{x}_S, \mathbf{x}_T, \hat{u}) \overset{(2)}{\leq} \mathrm{E}_{\mathbf{x}_S,\mathbf{x}_T}\, S(\mathbf{x}_S, \mathbf{x}_T, u_{opt}) \overset{(1)}{\approx} \mathrm{E}_{\mathbf{x}_S,\mathbf{x}_T}\, S(\mathbf{x}_S, \mathbf{x}_T, u) \tag{7}$$

Different assumptions used are written in pararentheses above each relation in the equation. ■

IV. Simultaneous Planning for Multi-Robot Assembly and Sensor Positioning

We now describe the use of task-specific uncertainty estimation within a planner for multi-robot assembly tasks. The planner we used is an extension of [18], which formulates multi-robot planning as a constraint satisfaction problem. The work of [18], however, does not reason about sensing and assumes perfect observability. We extend the planning algorithm to simultaneously solve the multi-robot planning *and* the sensor placement problem, by choosing the feasible plan with the most informative sensor pose. The resulting planner provides plans where the robots can perform the assembly, and the sensor is located to improve task success.

A. Problem

In our formulation, an assembly is a collection of simple parts at specific relative poses. Robots perform an *assembly operation* by grasping each part and bringing them together in space at these poses. An additional robot carries a sensor, e.g. a camera, to observe the parts and provide feedback to the assembly robots about the pose of each part. Formally, we define an assembly operation as a tuple $\langle \mathbf{A}, \mathbf{T}, \mathbf{L} \rangle$ where $\mathbf{A}$ is a set of assembly parts, $\mathbf{T} : \mathbf{A} \to SE(3)$ is a mapping from each part to its pose in the assembly, and $\mathbf{L}$ is a set of candidate poses for the sensor. A robot can grasp a part by placing its gripper at certain poses on the part. We assume we can compute a set of such poses, *grasps*. To bring a part to its pose in an assembly, a robot must plan a grasp, and a configuration for its base and arm. Similarly, we must plan the sensor robot configuration.

When the robots perform an assembly operation they must avoid colliding with each other, with the parts, and with other objects in the world. Moreover, they must choose a sensor position which provides useful feedback. Therefore, we formulate the problem of *simultaneous planning for multi-robot assembly and sensor positioning* as finding configurations for the robots such that they avoid collision while the sensor is positioned to provide the maximum probability of task success.

B. CSP Formulation

Given an assembly operation $o = \langle \mathbf{A}, \mathbf{T}, \mathbf{L} \rangle$, we can formulate the planning problem as a constraint satisfaction problem (CSP). A CSP is defined by a set of variables $\mathbf{V}$, a domain $\mathbf{D}(v)$ for each variable $v \in \mathbf{V}$ which specifies the set of possible values v can be assigned with, and a set of constraints specifying consistent assignments of values to variables. A solution to the CSP is an assignment of values to all the variables that is consistent with all the constraints. For our problem, we create one variable for the grasp of each input part. We use v_a to represent the variable corresponding to the grasp of assembly part $a \in \mathbf{A}$. The domain of the variable v_a is the set of robot base and arm configurations carrying the part at the pose $T(a)$ with a valid grasp. We discretize this possibly continuous set by sampling uniformly at a fine resolution. Finally, we define collision constraints between all the variables.

Backtracking search is a widely used and *complete* algorithm for solving CSPs [36]. It searches forward by assigning values to variables such that all assignments obey the constraints. If the algorithm cannot find a value for

a variable which obeys the constraints, it backtracks by undoing the most recent assignment. The search continues until an assignment is found for all variables. If there is no solution, backtracking search tries all combinations of value assignments. The worst-case time complexity of backtracking search is exponential in the number of variables. However, for the number of variables in our grasping formulation, solutions are found in tens of seconds on a CPU.

We use backtracking search to plan configurations for the robots grasping the parts and the robot carrying the sensor. However, while backtracking search finds non-colliding configurations, it does not reason about the quality of the sensor pose. To search for good sensor poses, we run backtracking many times and choose the solution with the highest success probability, as computed by Algorithm 2. We can replan if the maximal success probability is too low. However in our setup this was not needed.

V. Experiments

We consider mobile manipulation scenarios to evaluate the effectiveness of our approach. We demonstrate 3 real different scenarios using robots to execute the assembly, and explore another 4 in simulation. In simulation we also compute the utility of the sensor from a densely sampled set of sensor location and visualize resulting utility function. This allows us to gain insight on the planned locations beyond the specific chosen positions sampled by our planner. We note the assembly planner used in simulation, and the motion planner used for approximating task success are the same ones as used with the real robots.

A. Setup

We use a team of three or four KUKA Youbot single-arm robots. The robots are controlled via OpenRAVE, with a depth sensor mounted on one of the arms. We use VICON to track the robots' locations. Assembly parts are not marked, requiring the use of the depth sensor for pose correction, using our own ICP for pose estimation. The uncertainty estimation is implemented with OpenGL/CUDA. We sample poses with an prior distribution of an approximate Gaussian on the Lie algebra $se(3)$, and compute the expected scene statistics via OpenGL. We implement the approximate planner and collision checking via OpenRAVE/PQP. We use a truncated (at 5 cm) M-estimator for $d^2(\cdot,\cdot)$ for the likelihood term in Eq. 1, in both object localization and uncertainty estimation. In each assembly step, parts poses are positioned and corrected so that a local, linear, controller that can mate parts to each other. The system, with three robots is demonstrated in Figure 1(a).

B. Simulation Experiments

We first explore the uncertainty criteria computed in Algorithm 2 in several informative examples, to see what location would be chosen. In our simulations we focused on a basic step in assembly: a pair of objects are placed at assembly poses with varying amounts of prior uncertainty. Measuring the resulting criteria at various sensor poses around the objects allows us to visualize which locations would be deemed beneficial, and chosen by the overall planner. The sensor is aimed at the center of the two objects (placed at the origin). We found the exact aim to be less important, as long as the objects are fully seen in the sensor and capture the same field of view. We sample densely a 2D pose space for visualization purposes, unlike the actual use in an assembly planner. In the simulations we assume the sensor pose to be known with high certainty. This is done both since we focus on the uncertainty of the objects, but also since the sensor location is relatively well known (due to SLAM algorithms), as opposed to manipulated objects, whose location is less certain.

We measure the pose entropy and task success probability, as well as the effective sample size (N_{EFF}, [4]), shown as a function of the sensor's location in the (x, y) plane, given in meters. Pose entropy reduction demonstrates task-agnostic measures, and the task success probability represents a task-specific estimator for the assembly. The effective sample size shows when our sample set becomes impoverished.

For example, in Figure 4(a) the objects are a peg and a board with a hole. The peg has a handle, and a pie-slice of the peg is missing (also visible in Figure 3) which makes the peg's rotation important for successful insertion. We estimate the posterior uncertainty of the objects and the success probability of the task (inserting the peg) for several viewpoints around the peg (approximately at the height of lower third of the peg). We do this for two informative cases: In Figure 4(a) the uncertainty of the board is small, and the uncertainty of the peg is restricted to rotation about the peg's axis. The main informative areas are at the side and to the front of the section, where occlusions modify the range for a large set of pixels, or in front of the missing section, where many pixels change as a function of pose, but not at the back of the peg, where small rotations around the axis do not change the observed image. In Figure 4(b) we assume uncertainty in all 6 degrees of freedom. This time, task success favors the frontal view, which is less obvious in the entropy estimations that treat all degrees of freedom as equally important. This raises an important contrast with view planning based on visibility and occlusions, which may accept views (e.g.behind the peg) that cannot capture the rotation DOF and are hence suboptimal. Our method would prefer either frontal or side views, avoiding these limitations.

In Figure 4(c) we demonstrate the uncertainty reduction estimates for a pair of Lego blocks. In this case, areas in front of the blocks offer a lot of information due to the inserts' shape on the top of the block, as well as areas directly behind the blocks, which allow us to see how the two blocks align. Figure 4(d) demonstrates the uncertainty landscape around a star-shaped peg. Areas between the star's tips do not capture as much variability in the normals, at a close enough range and are less informative, whereas locations in front of the star' tip offer a more balanced distribution of normals and therefore more uncertainty reduction both in terms of the entropy, and task success. The average time to compute a single pose is around 60 seconds on a laptop with an NVIDIA GPU, depending on the number of samples (we took 80 samples for $\mathbf{x}_S$ and $\mathbf{x}_T$) and the type of approximate

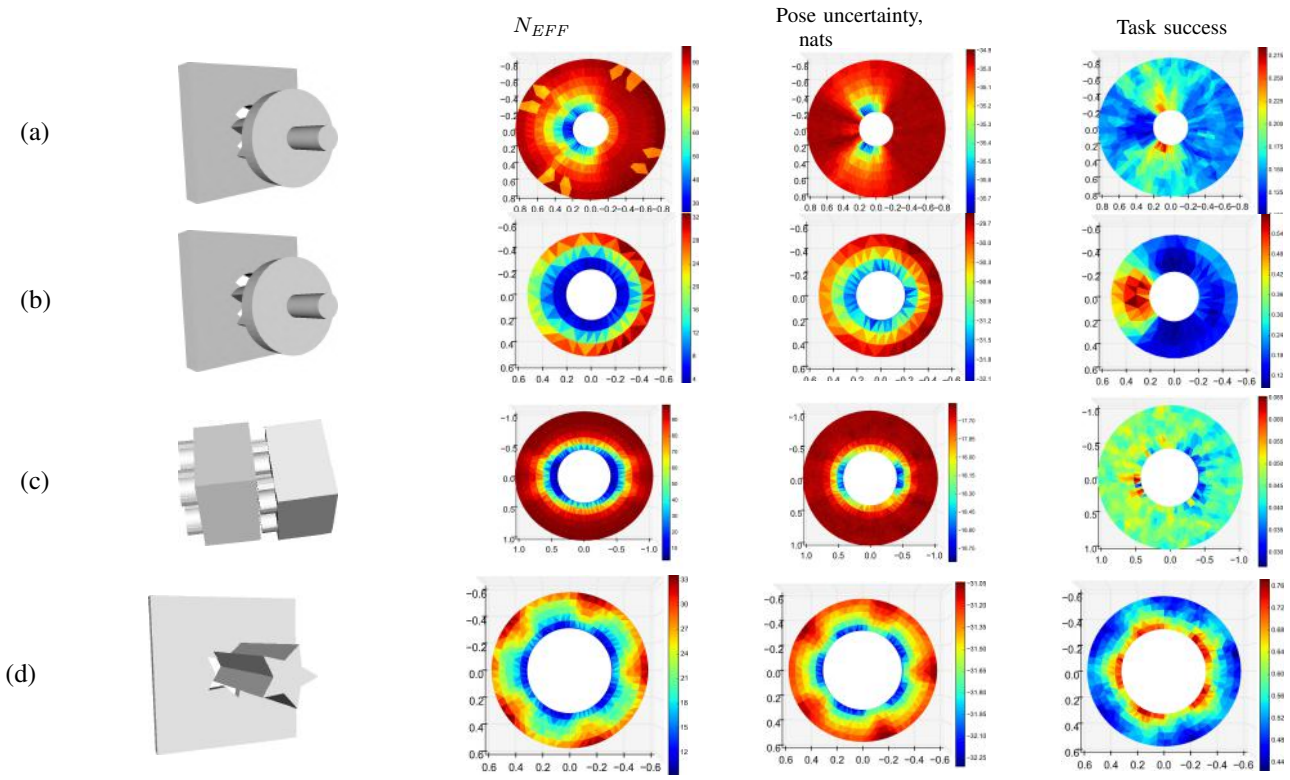


Fig. 4: Estimated utility at sampled along the 2D plane around the assembly objects. Locations are in metric coordinates, and sensor is aimed at the scene center. Left-to-right: objects configuration, N_{EFF}, pose uncertainty in nats, posterior task success estimate. Row **a)** - Peg and hole, insertion along the x axis, uncertainty only about peg rotation. Informative positions are along the sides of the peg, where occlusions modify the depth map considerably. Row **b)** - Peg and hole, insertion along the x axis, uncertainty in all 6 degrees of freedom. While the entropy is mostly reduced close to the peg, task certainty is increasing when the sensor is in front of the peg, which is the more informative view for this task. Row **c)** - Lego block assembly configuration, insertion along the x axis. While entropy is reduced almost uniformly since all DOFs are important, the Y-Z plane localization matters the most, giving preference to view from the front of the blocks (left side of the map), where the pins of the blocks allow better localization. Row **d)** - Star-shaped peg and hole, vertical insertion. The 5-fold symmetry is easily visible, with viewpoints close to the tips of the star being more informative.

planner used for success criteria. Most ($> 90\%$)of the current runtime is due to the CPU-based planner, leaving significant room for optimization, and we expect a GPU-based planner [32] to allow much greater efficiency. This makes it possible to run the planning with real robots, by picking a set of plan and compute the estimation quality for a selection of these. In the experiments we performed with only 2 parts (and no additional occlusions), good poses were twice as likely to succeed compared to bad ones, emphasizing the importance of correct sensor placement.

C. Robot Implementation

We used our approach as part of a robotic fixture-less assembly planner, as described in Sections IV and V-A. During planning we evaluate 32 candidate sensor locations, set in a circle around the assembled parts. In our experiments the system selects a plan where the sensor-bearing robot places the sensor in a pose that has an unobstructred view of the assembled objects. Using ICP we then reduce the part placement error to a compliant level (the initial standard deviation of > 2cm in object location prohibits compliant insertion). We demonstrate the following tasks with our system: 1) Placing a peg inside a hole in a board. 2) Inserting Jumbo Lego blocks into each other. 3) Stacking blocks on top of each other. The experiments are shown in Figure 5. In Figure 5(c)-(f) we show the rendering view of the planner for our tasks, demonstrating a successful location of the sensor with respect to the assembled parts.

VI. Conclusion

In this paper we demonstrated sensor planning based on elementary task primitives and their approximations. We showed how open-loop planners for these tasks allow us to estimate the quality of different observation poses, and yet can be run in reasonable time for on-line planning purposes in teams of real robots. We intend to explore additional types of elementary task primitives, as well as more rigorous definitions of the approximation encompassed by our framework.

Acknowledgment

This work was partially supported by the ONR (N00014-17-1-2072).

References

[1] S. Abrams, P. K. Allen, and K. A. Tarabanis. Dynamic sensor planning. In *ICRA*, pages 605–610. IEEE Computer Society Press, 1993.

[2] A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R. Rusu, and G. Bradski. CAD-model recognition and 6DOF pose estimation using 3D cues. In *ICCV Workshops*, pages 585–592, 2011.

[3] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *IJCV*, 1(4):333–356, 1988.

[4] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. Sig. Proc.*, 50(2):174–188, 2002.

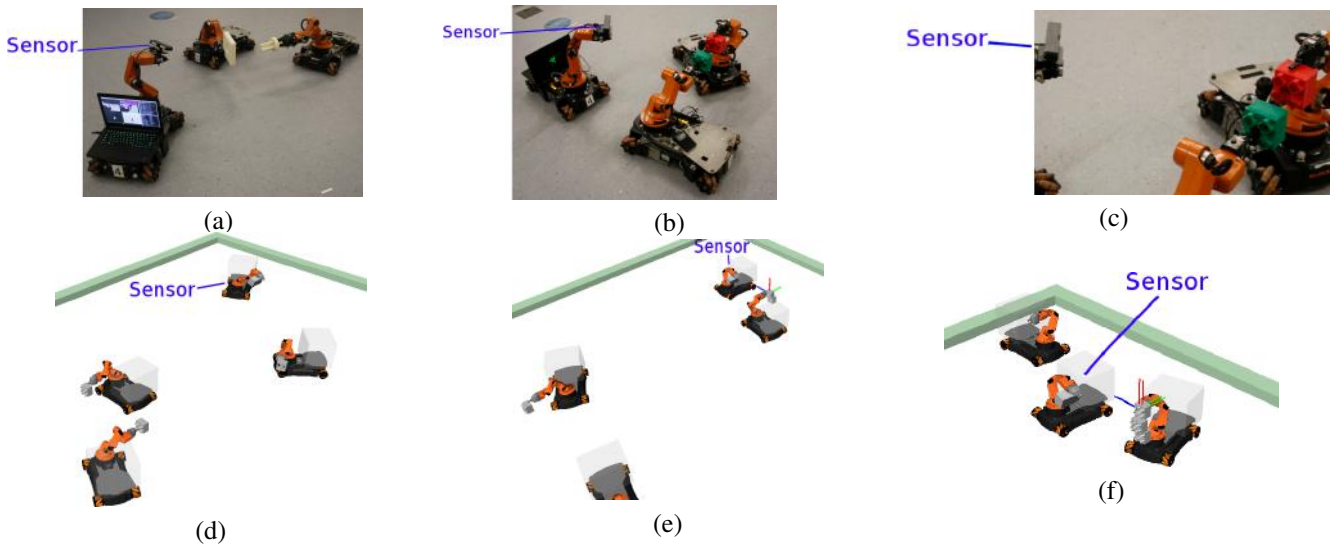


Fig. 5: Left to right: (a) Two robots placing peg in a board, (b-c) Two robots aligning two lego blocks for insertion. (d-f) Stacking three blocks, as seen from the planner view. (d) Robot placing first block. (e) Robot placing second block. (f) Third robot placing the third block. As can be seen, the planner selected sensor locations that are informative for the assembly steps.

[5] R. Bajcsy. Active perception. In *Proceedings of the IEEE*, volume 76, pages 966–1005. IEEE Press, Aug. 1988.

[6] E. Bar-Aviv and E. Rivlin. Functional 3D object classification using simulation of embodied agent. In *BMVC*, 2006.

[7] D. Berenson and S. S. Srinivasa. Grasp synthesis in cluttered environments for dexterous hands. In *HUMANOIDS*, 2008.

[8] D. Berenson, S. S. Srinivasa, and J. Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *IJRR*, 30(12):1435–1460, 2011.

[9] J.-M. Bernardo. Expected information as expected utility. *Ann. of Stat.*, 7:680–690, 1979.

[10] P. J. Besl and N. D. McKay. A method for registration of 3D shapes. *IEEE Trans. Pat. Anal. Mach. Intell.*, 14(2):239–256, 1992.

[11] B. Burns and O. Brock. Sampling-based motion planning with sensing uncertainty. In *ICRA*, pages 3313–3318, April 2007.

[12] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10:145–155, April 1992.

[13] C. Choi and H. I. Christensen. 3D pose estimation of daily objects using an RGB-D camera. In *IROS*, pages 3342–3349, 2012.

[14] D. Claes, P. Robbel, F. A. Oliehoek, K. Tuyls, D. Hennes, and W. van der Hoek. Effective approximations for multi-robot coordination in spatially distributed tasks. In *Int. Conf. on Auton. Agents and Multiagent Systems*, pages 881–890, 2015.

[15] C. Connolly. The determination of next best views. In *ICRA*, volume 2, pages 432–435, Mar 1985.

[16] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2006.

[17] H. Dang and P. K. Allen. Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *IROS*, 2012.

[18] M. Dogar, A. Spielberg, S. Baker, and D. Rus. Multi-robot grasp planning for sequential assembly operations. In *ICRA*, 2015.

[19] S. Foix, G. Alenya, and C. Torras. 3D sensor planning framework for leaf probing. In *IROS*, pages 6501–6506, Sept 2015.

[20] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *ISER*, 2010.

[21] H. Hino and N. Murata. Information estimators for weighted observations. *Neural Networks*, 46:260 – 275, 2013.

[22] L. P. Kaelbling and T. Lozano-Pérez. Integrated task and motion planning in belief space. *IJRR*, 32(9-10), 2013.

[23] J. Kim, K. Iwamoto, J. J. Kuffner, Y. Ota, and N. S. Pollard. Physically based grasp quality evaluation under pose uncertainty. *Trans. on Robotics*, 29(6):1424–1439, 2013.

[24] M. Krainin, B. Curless, and D. Fox. Autonomous generation of complete 3D object models using next best view manipulation planning. In *ICRA*, 2011.

[25] K. Lai, L. Bo, X. Ren, and D. Fox. Sparse distance learning for object recognition combining RGB and depth information. In *ICRA*, pages 4007–4013, 2011.

[26] J. J. Leonard and D. H. Whyte. Mobile robot localization by tracking geometric beacons. *Trans. on Robotics and Automation*, 7(3), 1991.

[27] D. V. Lindley. On a Measure of the Information Provided by an Experiment. *The Annals of Math. Stat.*, 27(4):986–1005, 1956.

[28] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *ICML*, 1995.

[29] N. J. Mitra, N. Gelfand, H. Pottmann, and L. Guibas. Registration of point cloud data from a geometric optimization perspective. In *Eurographics Symp. on Geom. Proc.*, pages 23–31, 2004.

[30] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Nat. Conf. on Artificial Intelligence*, pages 593–598, 2002.

[31] K. P. Murphy. A survey of POMDP solution techniques. Technical report, 2000.

[32] J. Pan and D. Manocha. Gpu-based parallel collision detection for fast motion planning. *IJRR*, page 0278364911429335, 2011.

[33] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *J. Visual Communication and Image Representation*, 25(1):148–164, Jan 2014.

[34] G. Rosman, D. Rus, and J. W. F. III. Information-driven adaptive structured-light scanners. In *CVPR*, pages 874–883, 2016.

[35] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *JAIR*, 32:663–704, 2008.

[36] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. 2nd edition, 2003.

[37] G. Shani, J. Pineau, and R. Kaplow. A survey of point-based POMDP solvers. *Aut. Agents and Multi-Agent Systems*, 27(1):1–51, July 2013.

[38] K. Sricharan and A. O. Hero. Ensemble weighted kernel estimators for multivariate entropy estimation. In *NIPS*, pages 566–574. 2012.

[39] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[40] N. Vahrenkamp, E. Kuhn, T. Asfour, and R. Dillmann. Planning multi-robot grasping motions. In *HUMANOIDS*, 2010.

[41] S. Wenhardt, B. Deutsch, J. Hornegger, H. Niemann, and J. Denzler. An information theoretic approach for next best view planning in 3-D reconstruction. In *ICPR*, volume 1, pages 103–106, 2006.

[42] P. M. Will and D. D. Grossman. An experimental system for computer controlled mechanical assembly. *IEEE T-Computers*, 24(9):879–888, 1975.

[43] J. L. Williams, J. W. Fisher III, and A. S. Willsky. Performance guarantees for information theoretic active inference. *JMLR*, 2:620–627, 2007.