BEACON: Belief-Aware Replanning for Safe Online Motion Planning

Ishita Banerjee*, Maggie Bai*, Soptorshi Ghosh, Rhea Senan, Ayushi Mehrotra
California Institute of Technology
Pasadena, CA, USA
{ibanerje, mbai, srai, rsenan, amehrotra}@caltech.edu

***Abstract*—Robots operating in partially known environments must balance goal-directed progress with uncertainty about newly revealed obstacles and the consequences of interacting with them. This challenge is especially important in contact-rich navigation, where contact can provide useful information and open shorter paths while also consuming energy and increasing interaction risk. We present BEACON, a belief-aware replanning framework that maintains per-obstacle beliefs over interaction class and uses those beliefs to guide local action selection during online motion planning. BEACON combines semantic risk estimates, reactive replanning, and physically grounded contact feedback to decide when to push and when to detour. We evaluate BEACON across 400 procedurally generated benchmark episodes spanning sparse, cluttered, collision-required, and collision-shortcut environments. At the default configuration, BEACON achieves a 96.75% success rate, an average path length of 5.053 m, and an average of 132.16 steps. Although D* Lite has a slightly higher default success rate of 97.25%, BEACON produces substantially shorter trajectories, reducing average path length by 39.8% relative to D* Lite. Under the best configuration selected from the hyperparameter sweep, both D* Lite and BEACON reach 100.00% success, while BEACON uses slightly less average energy. These results suggest that belief-aware contact reasoning improves efficiency in settings where safe exploration and online replanning must be balanced against resource use. Code: https://anonymous.4open.science/r/BEACON-78C0.**

I. Introduction

Exploration is a central capability for autonomous robots operating in partially known environments. A robot must gather information about its surroundings, adapt as new obstacles are revealed, and make progress toward its goal under safety and resource constraints. These challenges are especially sharp in contact-rich navigation, where the robot must decide whether to push an obstacle out of the way or route around it. This push-versus-avoid decision creates an exploration-exploitation tradeoff: contact can reveal useful information and open shorter paths, while unsafe or unnecessary contact can waste energy, damage objects, or delay task completion.

This setting aligns with the broader view of exploration as a unifying problem across robotics, reinforcement learning, and search. In robotics, exploration is constrained by physical interaction, safety, and limited sensing. In search, exploration appears through replanning, heuristic guidance, and online updates as new parts of the state space become known. In reinforcement learning, exploration is often framed through information gathering, uncertainty reduction, and the tradeoff between short-term cost and long-term value. Contact-rich navigation sits at the intersection of these perspectives because each contact attempt is both an action in the world and an observation about the world.

Recent contact-aware planners use semantic labels from vision-language models, semantic segmentation, or obstacle classifiers to decide when contact is acceptable [1], [2]. These systems show that semantic information can improve navigation in cluttered environments. However, many contact-aware methods commit to an interaction model when an obstacle is first observed. The resulting labels or costs are then used throughout execution, even as the robot receives new evidence through motion, failed pushes, successful displacements, and changes in local geometry. This limits the planner's ability to use exploration as a mechanism for improving its own world model.

Classical motion planning provides strong geometric foundations for navigation in partially known spaces. Sampling-based and kinodynamic planners reason about feasible motion under geometric and dynamic constraints [3]–[5], while reactive and incremental planners such as Bug, Bug2, and D* Lite support online navigation as obstacles are discovered [6], [7]. These methods generally treat obstacles as collision constraints, which limits their ability to reason about informative or useful contact. Recent contact-aware and semantic planners relax this assumption by assigning obstacle-dependent traversal or contact costs [1], [2], [8], [9]. In most of these systems, semantic information is extracted at observation time and then used as a fixed cost signal.

Belief reasoning offers a natural mechanism for exploration under changing uncertainty. Belief-space planners track pose uncertainty for localization-aware navigation [10], [11], and partially observable task-and-motion planners reason over uncertain object interactions and risks [12]. Other online semantic planning methods update maps or policies when new observations change semantic confidence [13], [14]. Contact feedback is especially relevant because it can directly reveal whether an obstacle is movable, fragile, or unsafe to push [15]. These lines of work motivate a planner that treats physical interaction as an information source and uses belief updates to guide future action selection.

We argue that the push-versus-avoid decision should be framed as safe, resource-aware exploration under evolving semantic uncertainty. The relevant decision features include

the relative costs of pushing and detouring, proximity to the goal, stall state, predicted corridor gain from displacing an obstacle, remaining battery, and available safety margin. These features change as the robot explores. A planner that updates its interaction beliefs during execution can use contact outcomes to refine future decisions and adapt its behavior to the local structure of the environment.

We present BEACON, a belief-aware replanning framework for online motion planning. BEACON maintains a local semantic risk belief for newly revealed obstacles, updates that belief as execution provides new evidence, and uses it to score candidate actions during replanning. The planner combines belief-weighted risk evaluation with a reactive local action-selection rule that trades off goal progress, geometric and directional safety, and resource expenditure. In the context of exploration, BEACON provides a concrete case study of how search-based replanning, semantic uncertainty, and physically grounded feedback can be integrated for safe adaptation in cluttered environments. A fuller discussion of related work appears in Appendix C.

Our contributions are three-fold. (i) We integrate per-obstacle belief over interaction class with reactive local action selection in a battery-constrained partially observable navigation setting. (ii) We design a benchmark of four procedurally generated scene families, sparse, cluttered, collision-required, and collision-shortcut, that exposes when exploratory contact and belief updates matter. (iii) We evaluate BEACON against geometric and contact-aware baselines, showing that belief-aware replanning achieves short paths and competitive success while preserving a resource-aware contact policy.

II. Methods

A. Preliminaries

Let the workspace be the bounded planar region $\Omega = [0,6] \times [0,6] \subset \mathbb{R}^2$. The robot starts at $x_0 \in \Omega$ and must reach a goal $x_g \in \Omega$, with $\|x_0 - x_g\|_2 \geq r_R$ where the robot is modeled as a disk of radius $r_R > 0$, treated as a tunable collision-checking hyperparameter. The environment contains a finite obstacle set $\mathcal{O} = \{O_1, \ldots, O_N\}$, where each obstacle O_i is described by a polygonal footprint at position p_i with geometry g_i, volume V_i, material density ρ_i, and friction coefficient μ_i. Obstacle shapes include circles, rectangles, triangles, trapezoids, parallelograms, pentagons, and hexagons. Obstacle mass follows directly from density and volume as $m_i = \rho_i V_i$.

At time t, the robot state is (x_t, v_t, B_t), where $x_t \in \Omega$ is position, $v_t \in \mathbb{R}^2$ is velocity, and $B_t \in [0, B_0]$ is the remaining battery, normalized to $B_0 = 1$. Battery depletes with travel:

$$B_{t+1} = \max(0,\ B_t - \eta \|x_{t+1} - x_t\|_2)\,, \quad \eta = \frac{B_0}{\operatorname{diam}(\Omega)}.$$

The robot operates under partial observability and senses only obstacles within radius $r_s > 0$ of its body, where r_s is also treated as a sensing hyperparameter. Thus the observed obstacle set

$$\mathcal{O}_t = \big\{ O_i \in \mathcal{O} \ :\ \operatorname{dist}\big(\mathcal{B}(x_t, r_R), O_i\big) \leq r_s \big\}$$

grows incrementally as the robot moves. Each obstacle additionally carries a latent interaction class $c_i \in \mathcal{C}$ that determines the cost and feasibility of physical interaction. The class is unobserved at first detection and is the central uncertainty BEACON reasons over.

B. Local Semantic Risk Belief

For each observed obstacle $O_i \in \mathcal{O}_t$, BEACON maintains a categorical belief over the latent interaction class:

$$b_i^{(t)}(c) = \Pr(c_i = c \mid \mathcal{H}_t)\,, \qquad c \in \mathcal{C},$$

where $\mathcal{H}_t$ denotes the history of observations and contact outcomes accumulated up to time t. The class space $\mathcal{C}$ covers a small set of qualitative interaction modes that differ in resistance to motion and downstream cost. On first observation, the belief is initialized from a class prior π_0 computed from the obstacle's perceived geometric and material features:

$$b_i^{(0)}(c) = \pi_0(c \mid g_i, \rho_i, \mu_i).$$

The prior π_0 is not learned. Instead, if an obstacle provides semantic probabilities, BEACON normalizes and uses them directly; otherwise, the belief is initialized as a one-hot distribution on the obstacle's ground-truth semantic label:

$$\pi_0(c \mid g_i, \rho_i, \mu_i) = \begin{cases} \dfrac{s_i(c)}{\sum_{c' \in \mathcal{C}} s_i(c')} & \text{scores available,} \\ 1 & c = c_i^{\text{true}}, \\ 0 & \text{otherwise.} \end{cases}$$

C. Belief Update from Contact Evidence

When the robot applies a force to obstacle O_i, it observes a binary outcome $e \in \{\texttt{moved}, \texttt{not moved}\}$ reflecting whether the obstacle was displaced. Each interaction class induces a likelihood $P(e \mid c)$ that determines how diagnostic the observation is. The belief is updated by Bayes' rule,

$$b_i^{(t+1)}(c) \ \propto\ P(e \mid c)\, b_i^{(t)}(c),$$

followed by normalization so that $\sum_{c \in \mathcal{C}} b_i^{(t+1)}(c) = 1$. Because contact outcomes are highly informative about interaction class, even a single contact event can produce a sharp change in belief, which in turn changes the local action-selection objective described below.

D. Force and Energy Model

Whether a candidate push is feasible depends on a simple Coulomb-friction model. The minimum tangential force required to initiate motion of obstacle O_i on the supporting surface is

$$F_i = \mu_i\, m_i\, g = \mu_i\, \rho_i\, V_i\, g,$$

where g is gravitational acceleration. The robot can apply force up to a limit $F_{\text{robot}}^{\max}$, so a push is geometrically feasible only if $F_i \leq F_{\text{robot}}^{\max}$. When feasible, displacing O_i by distance d costs energy

$$E_i = F_i\, d.$$

This force-energy model couples obstacle properties to the resource term of the action-selection objective and ensures that push decisions trade off against remaining battery in a physically meaningful way.

E. Belief-Weighted Action Selection

At each control step, BEACON selects the next state from a finite candidate set Q_t of locally feasible candidates by minimizing a composite objective:

$$q_t^\star = \arg\min_{q \in Q_t} J(q),$$
$$J(q) = J_{\text{pos}}(q) + J_{\text{risk}}(q) + J_{\text{vel}}(q) + J_{\text{res}}(q).$$

The four terms encode goal progress, belief-weighted interaction risk, velocity alignment after potential reflection, and resource expenditure, respectively.

a) Position term.: The position term penalizes remaining goal distance modulated by heading deviation:

$$J_{\text{pos}}(q) = w_p\big(\|x(q) - x_g\|_2 + \theta_{\text{head}}(q)\big),$$

where $\theta_{\text{head}}(q)$ is the angle between the candidate velocity and the goal direction.

b) Belief-weighted risk term.: The risk term takes the expectation of a per-class risk function under the current belief over the nearest observed obstacle:

$$\begin{aligned} J_{\text{risk}}(q) &= w_r\, \mathbb{E}_{c \sim b_{i^\star(q)}^{(t)}}\big[R(c)\big] \\ &= w_r \sum_{c \in \mathcal{C}} b_{i^\star(q)}^{(t)}(c)\, R(c), \end{aligned}$$

where $i^\star(q)$ indexes the obstacle nearest to candidate q and $R(c)$ is the per-class interaction risk. This is the term that distinguishes BEACON from fixed-label contact-aware planners: as the belief sharpens with contact evidence, the expected risk for each candidate updates accordingly.

c) Velocity term.: The velocity term measures how favorably the robot's motion aligns with the goal after reflection from the nearest obstacle's contact normal:

$$J_{\text{vel}}(q) = w_v\big(1 - \cos\theta_{\text{post}}(q)\big),$$

where $\theta_{\text{post}}(q)$ is the angle between the post-reflection direction and the goal direction. This term penalizes candidates that, on contact, would deflect the robot away from the goal.

d) Resource term.: The resource term combines travel distance, traversal time, and contact energy:

$$J_{\text{res}}(q) = w_d\, d(q) + w_t\, T(q) + w_e\, \mathbf{1}_{\text{push}}(q)\, E_{i^\star(q)},$$

where $d(q) = \|x(q) - x_t\|_2$, $T(q) = d(q)/\max(\|v(q)\|_2, \varepsilon)$, and $\mathbf{1}_{\text{push}}(q)$ indicates whether the candidate involves contact-induced displacement.

e) Battery-dependent reweighting.: As battery depletes, the planner shifts its priorities toward energy-efficient motion. We let $\beta_t = B_t/B_0 \in [0,1]$ and reweight the risk and velocity terms by

$$w_r(\beta_t) = w_r^{\text{scale}}\,\beta_t, \qquad w_v(\beta_t) = w_v^{\text{floor}} + w_v^{\text{range}}(1 - \beta_t),$$

so that risk aversion decreases and velocity-alignment pressure increases as $\beta_t \to 0$. The remaining weights (w_p, w_d, w_t, w_e) are constant.

$$\begin{aligned} c(u,v,t) = c_{\text{geom}}(u,v) &+ \lambda_{\text{occ}}\, c_{\text{occ}}(v) \\ &+ \lambda_{\text{sem}}\, c_{\text{sem}}(v,t), \end{aligned}$$

where $\lambda_{\text{occ}} > 0$ weights occupancy cost and $\lambda_{\text{sem}} > 0$ weights semantic cost. The discretization step $\Delta > 0$ and step duration $\tau > 0$ are also treated as planner hyperparameters. For push-versus-detour comparison, the controller additionally uses two decision hyperparameters b_{push} and m_{detour}.

F. Push-versus-Detour Decision

For each obstacle encountered along the path, BEACON evaluates the two qualitatively distinct candidate actions, a_{push} and a_{detour}, under the current belief. Let

$$C_{\text{push}} = J(a_{\text{push}}), \qquad C_{\text{detour}} = J(a_{\text{detour}}).$$

The robot pushes if $C_{\text{push}} < C_{\text{detour}}$ and detours otherwise. In the implemented controller, this comparison is parameterized by the push-bias and detour-margin hyperparameters, so the effective decision rule is

$$\text{push if } C_{\text{push}} - b_{\text{push}} < C_{\text{detour}} + m_{\text{detour}},$$

and detour otherwise. Because the risk term J_{risk} is a belief-weighted expectation, the push cost adapts to evidence: as belief mass concentrates on `movable` classes for an obstacle, C_{push} decreases, and the planner becomes more willing to push that specific obstacle even when the global cost weights are unchanged.

G. Chain-Reaction Dynamics

Because pushed obstacles can collide with neighboring obstacles, BEACON simulates the cascade of induced displacements before committing to a push. When O_i is displaced into a neighbor O_j, the planner evaluates whether O_j itself is feasible to displace under the same force model. The cascade $O_i \to O_j \to O_k \to \cdots$ terminates when the next obstacle in the chain is infeasible or no further contact occurs. The resulting set of post-push positions is recorded, while the original configuration is preserved for visualization and ablation comparisons.

H. Replanning Trigger

BEACON triggers replanning under either of two conditions. The first is a belief change exceeding a threshold:

$$\big\|b_i^{(t+1)} - b_i^{(t)}\big\|_1 > \epsilon_b \quad \text{for some } O_i \in \mathcal{O}_t,$$

where the ℓ_1 distance over the categorical distribution measures how much new evidence has shifted the planner's interaction model. The second is a map change: any obstacle whose position is updated through a push action invalidates the previous local cost map. In either case, the underlying D* Lite layer recomputes the path on the updated cost map, and local action selection resumes from the new state.

I. Execution Loop

At each control step, BEACON (1) advances along the current planned path, (2) on encountering an obstacle, evaluates push and detour candidates under the current belief and force constraints, (3) selects the action minimizing J, (4) executes the resulting motion or push, (5) updates the obstacle map and the affected obstacle's belief from the observed outcome, and (6) triggers replanning if the belief or map change conditions are met. This loop allows BEACON to revise both its physical interaction strategy and its internal interaction model as execution proceeds, resolving ambiguity through directed contact while maintaining global progress toward the goal.

III. Experimental Setup

A. Baselines

We compare BEACON against four baselines: Bug, Bug2, D* Lite, and a contact-aware semantic baseline. Bug [6] and Bug2 [6] are reactive geometric planners that alternate between direct goal pursuit and obstacle boundary following, with Bug2 additionally tracking an explicit start-to-goal reference line. D* Lite [7] is an incremental graph-search planner that maintains a shortest-path estimate over a discretized workspace and updates it efficiently as new obstacle information is revealed, providing the strongest classical search-based comparison. Together, these three planners establish a geometric reference floor: they cover the spectrum from purely reactive local strategies to full incremental search, and a contact-aware planner that fails to outperform this floor would offer no value over collision-free planning. The contact-aware semantic baseline shares BEACON's local action-selection structure but uses fixed interaction weights rather than belief-aware scoring, isolating the contribution of belief reasoning from contact-awareness alone. Full descriptions of each baseline appear in Appendix A. Direct comparison against published learned contact-aware planners such as IMPACT [1] and the reinforcement learning baselines in Bench-Push [8] requires training infrastructure and benchmark integration beyond the scope of this workshop submission and is left for future work.

B. Environments

We evaluate planner performance across four families of procedurally generated environments designed to capture distinct navigation regimes. Scenes are generated programmatically using a fixed random seed to ensure reproducibility. Each scene is serialized to JSON, storing obstacle geometry, pose, semantic class, and physical parameters for downstream evaluation and cross-platform consistency.

a) 2D environments.: All 2D scenes are generated within the bounded 6×6 workspace defined in Section III-A and constructed using `Shapely` for geometric operations and `Matplotlib` for visualization. Obstacles are sampled from a diverse set of polygonal primitives, including circles (approximated via buffered points), rectangles, triangles, trapezoids, parallelograms, pentagons, and hexagons. Each obstacle is randomly scaled, rotated, and placed using rejection sampling to enforce non-overlap while allowing boundary contact. A clearance buffer is maintained around start and goal positions.

Each obstacle is assigned a semantic class from $\{\text{safe}, \text{movable}, \text{fragile}\}$, which governs interaction behavior. Scene families are defined as follows: *sparse* environments contain relatively few obstacles and broad free space, primarily testing baseline geometric navigation; *cluttered* environments feature dense obstacle configurations that induce frequent short-horizon trade-offs between path efficiency and interaction risk; *collision-required* environments include a structured, workspace-spanning barrier composed of tightly packed obstacles, making collision-free traversal unlikely and requiring intentional interaction with movable objects; and *collision-shortcut* environments admit collision-free solutions but include strategically placed movable obstacles along the nominal shortest path, allowing shorter routes through controlled interaction.

b) 3D environments.: To extend evaluation beyond planar navigation, we construct a corresponding set of 3D environments implemented in `PyBullet`. These environments preserve the structural characteristics of the 2D families while introducing volumetric geometry and physics-based interactions. The workspace is defined as a bounded region in $\mathbb{R}^3$, populated with primitive rigid bodies including spheres, boxes, cylinders, and capsules. Obstacle placement follows the same procedural logic as the 2D generator, with structured clutter patterns (e.g., extended barriers in collision-required scenes) and randomized placement elsewhere.

Each obstacle is assigned a semantic class consistent with the 2D setting, with class-dependent physical properties. *Movable* objects are assigned nonzero mass and are dynamically simulated using rigid-body physics. *Safe* objects are typically static or low-penalty contacts. *Fragile* objects are associated with low damage thresholds, implemented via a contact-force model in which interactions exceeding a predefined force threshold incur a penalty and may mark the object as damaged or broken. Mass is computed as the product of material density and geometric volume, and friction coefficients govern contact behavior. These properties are consistent with the force-energy interaction model described in Section III-D.

C. Benchmark Protocol

The benchmark consists of 400 episodes per planner, with 100 scenes drawn from each environment family. All scenes are procedurally generated using fixed seeds to ensure reproducibility while maintaining diversity across trials.

We report success rate, average number of steps, average path length, average number of contact events, and average number of sensed obstacles. Formal definitions of all evaluation metrics are provided in Appendix B.

D. Hyperparameter Sweep

We sweep the planner hyperparameters that shape the behavior of the contact-aware planners and report the selected configuration in Table I. The swept parameters include robot radius, energy weight, occupancy weight, semantic weight,

TABLE I: Planner hyperparameter sweep. Swept on the cluttered subset; geometric baselines have no swept parameters.

Parameter	**Default**	**Sweep**	**Best**
Robot radius r_R	0.12 m	$\{0.12, 0.45\}$ m	0.15 m
Energy weight w_e	1.0	$\{0.5, 1.5\}$	1.0
Occupancy weight λ_{occ}	5.0	$\{2.5, 5.0, 7.5\}$	5.0
Semantic weight λ_{sem}	2.5	$\{1.5, 3.5\}$	2.5
Push bias b_{push}	0.35	$\{0.20, 0.50\}$	0.35
Detour margin m_{detour}	0.5	$\{0.25, 0.75\}$	0.50

TABLE II: Overall results across 400 episodes at the default benchmark configuration.

Planner	Success Rate	Avg. Steps	Avg. Path (m)
Bug	82.25%	180.47	9.604
Bug2	91.25%	185.97	8.055
D* Lite	97.25%	159.67	8.387
Baseline Semantic	89.25%	214.82	7.898
BEACON	**96.75%**	**132.16**	**5.053**

push bias, and detour margin. Geometric baselines (Bug, Bug2, and D* Lite) do not use these contact-aware hyperparameters and are reported at their default settings. The best-configuration comparison appears in Table III.

IV. Results

Table II summarizes the default benchmark results across 400 episodes. BEACON reaches a 96.75% success rate, compared with 82.25% for Bug, 91.25% for Bug2, 97.25% for D* Lite, and 89.25% for the fixed semantic baseline. D* Lite achieves the highest default success rate, exceeding BEACON by 0.50 percentage points. BEACON achieves the strongest path-efficiency result, with an average path length of 5.053 m. This is a 39.8% reduction relative to D* Lite and a 36.0% reduction relative to the fixed semantic baseline.

BEACON also requires fewer executed steps than all baselines at the default configuration. It averages 132.16 steps, compared with 180.47 for Bug, 185.97 for Bug2, 159.67 for D* Lite, and 214.82 for the fixed semantic baseline. This step reduction indicates that belief-aware contact reasoning helps the planner make more direct progress once it has enough local information to choose between pushing and detouring.

Table III reports the best configuration selected from the hyperparameter sweep. Under these settings, D* Lite and BEACON both reach 100.00% success. BEACON has slightly lower average total energy, with $\bar{E} = 24.447$ compared with $\bar{E} = 24.856$ for D* Lite. The energy gap is small, but it is consistent with the default benchmark trend that BEACON maintains high task success while selecting more efficient trajectories.

V. Discussion

The results show that BEACON is strongest on trajectory efficiency while remaining competitive on success. At the default configuration, D* Lite has the highest success rate by a narrow margin, while BEACON produces the shortest paths and the fewest executed steps. This pattern suggests that belief-aware contact reasoning is most useful when the planner can trade local interaction risk for more direct progress through the environment.

TABLE III: Best-configuration results from the hyperparameter sweep. Success rate (SR) and average total energy per episode ($\bar{E}$, normalized units), aggregated over 400 episodes per planner. Selection criterion: highest SR with $\bar{E}$ as a tiebreaker.

Planner	SR (%)	$\bar{E}$
Bug	44.44	11.493
Bug2	22.22	5.657
D* Lite	100.00	24.856
BEACON	**100.00**	**24.447**

The comparison with the fixed semantic baseline is especially important. The fixed semantic baseline has access to semantic costs, yet it reaches lower success, takes more steps, and produces longer paths than BEACON. This gap indicates that semantic contact awareness benefits from online adaptation. BEACON uses execution evidence to adjust how it evaluates contact, which helps avoid overly conservative detours and inefficient local choices.

The best configuration results show that BEACON can match the strongest classical baseline in success while using slightly less energy. D* Lite remains a strong method because it replans efficiently as new geometry is revealed. BEACON adds a contact-aware decision layer that can improve path and energy efficiency when interaction with obstacles is useful. The results therefore support the value of belief-aware replanning as a mechanism for safe, resource-aware exploration in contact-rich environments.

Several limitations remain. The experiments are conducted in simulation, and the benchmark scenes are generated from the same procedural families used during development. The current evaluation also focuses on aggregate success, path length, step count, and energy. Future work should test BEACON on physical robots, compare against learned contact-aware planners, and add more detailed measurements of contact safety, object damage, and uncertainty calibration.

VI. Conclusion

We presented BEACON, a belief-aware replanning framework for contact-rich online motion planning. BEACON maintains per-obstacle beliefs over interaction class and uses those beliefs to guide push-versus-detour decisions under safety and resource constraints. Across 400 benchmark episodes, BEACON achieves 96.75% success, the shortest average path length, and the fewest average steps among the evaluated planners at the default configuration. Under the selected best configuration, BEACON reaches 100.00% success and slightly lower average energy than D* Lite. These results show that belief-aware replanning can support safe and efficient explo-

ration when robots must reason about both geometry and uncertain physical interaction.

REFERENCES

[1] Yiyang Ling, Karan Owalekar, Oluwatobiloba Adesanya, Erdem Bıyık, and Daniel Seita. IMPACT: Intelligent motion planning with acceptable contact trajectories via vision-language models. *arXiv preprint arXiv:2503.10110*, 2025.

[2] Marco Masannek, Rolf Schmidt, Andreas Deinlein, Thorsten Gecks, Stefan May, and Andreas Nüchter. Semantic-aware obstacle tracking and avoidance for ceiling-mounted healthcare robots. In *Proceedings of the IEEE/SICE International Symposium on System Integration (SII)*, 2026.

[3] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[4] Oren Salzman, Michael Hemmer, Barak Raveh, and Dan Halperin. Motion planning via manifold samples. In *Proceedings of the 19th Annual European Symposium on Algorithms (ESA)*, pages 493–505, 2011.

[5] Dustin J. Webb and Jur van den Berg. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5054–5061, 2013.

[6] Vladimir J. Lumelsky and Alexander A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2(1):403–430, 1987.

[7] Sven Koenig and Maxim Likhachev. D* lite. In *Proceedings of the 18th AAAI Conference on Artificial Intelligence (AAAI)*, pages 476–483, 2002.

[8] Ninghan Zhong, Steven Caro, Megnath Ramesh, Rishi Bhatnagar, Avraiem Iskandar, and Stephen L. Smith. Bench-push: Benchmarking pushing-based navigation and manipulation tasks for mobile robots. *arXiv preprint arXiv:2512.11736*, 2025.

[9] Szymon Bartoszyk, Patryk Kasprzak, and Dominik Belter. Terrain-aware motion planning for a walking robot. In *Proceedings of the 11th International Workshop on Robot Motion and Control (RoMoCo)*, pages 29–34, 2017.

[10] Ali akbar Agha-mohammadi, Suman Chakravorty, and Nancy M. Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *The International Journal of Robotics Research*, 33(2):268–304, 2014.

[11] Ali akbar Agha-mohammadi, Saurav Agarwal, Sung-Kyun Kim, Suman Chakravorty, and Nancy M. Amato. SLAP: Simultaneous localization and planning under uncertainty via dynamic replanning in belief space. *IEEE Transactions on Robotics*, 34(5):1195–1214, 2018.

[12] Aidan Curtis, George Matheos, Nishad Gothoskar, Vikash Mansinghka, Joshua Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Partially observable task and motion planning with uncertainty and risk awareness. *arXiv preprint arXiv:2403.10454*, 2024.

[13] Po-Chen Ko, Hung-Ting Su, Ching-Yuan Chen, Jia-Fong Yeh, Min Sun, and Winston H. Hsu. Context-aware replanning with pre-explored semantic map for object navigation. In *Proceedings of the 8th Conference on Robot Learning (CoRL)*, 2024.

[14] Azizollah Taheri and Derya Aksaray. Motion planning under temporal logic specifications in semantically unknown environments. *arXiv preprint arXiv:2511.03652*, 2025.

[15] Brad Saund, Sanjiban Choudhury, Siddhartha Srinivasa, and Dmitry Berenson. The blindfolded robot: A Bayesian approach to planning with contact feedback. In *Proceedings of the International Symposium on Robotics Research (ISRR)*, 2019.

APPENDIX A
BASELINE PLANNER DESCRIPTIONS

We compare BEACON against four baselines that cover purely reactive geometric strategies, incremental graph search, and a contact-aware planner that lacks belief reasoning. We adopt the notation from the main text: $x_t \in \Omega$ is the robot position at time t, $x_g \in \Omega$ the goal, $\mathcal{O}_t \subseteq \mathcal{O}$ the set of observed obstacles within sensing radius r_s, and $\mathcal{C}_{\text{free}}^{(t)} = \Omega \setminus \bigcup_{O \in \mathcal{O}_t} O$ the locally known free space. The straight-line direction toward the goal is $\hat{u}_g(x) = (x_g - x)/\|x_g - x\|_2$, and the boundary of the obstacle currently in contact is ∂O_{hit}.

A. Bug1

Bug1 [6] is a reactive planner that uses only local contact information and the bearing to the goal. It alternates between two modes. In *motion-to-goal* mode, the robot moves along $\hat{u}_g(x_t)$ until either reaching x_g or contacting an obstacle at a hit point x_H. In *boundary-following* mode, the robot circumnavigates ∂O_{hit} recording, for every visited boundary point x, the distance $d(x, x_g) = \|x - x_g\|_2$. After completing one full traversal of the boundary, it returns to the leave point

$$x_L \in \arg\min_{x \in \partial O_{\text{hit}}} d(x, x_g),$$

and resumes motion-to-goal from x_L. Bug1 maintains no global map and uses only local geometric collision information from $\mathcal{O}_t$, so its update rule depends solely on $(x_t, x_g, \partial O_{\text{hit}})$. It is complete in finite-obstacle environments but its worst-case path length scales with the sum of obstacle perimeters encountered.

B. Bug2

Bug2 [6] replaces the full-perimeter scan of Bug1 with an explicit reference line, the m-line,

$$\mathcal{M} = \{x_0 + \lambda(x_g - x_0) : \lambda \in [0, 1]\},$$

connecting the start to the goal. The planner moves along $\mathcal{M}$ until contact at a hit point x_H. It then follows ∂O_{hit} until reaching the first point

$$x_L \in \partial O_{\text{hit}} \cap \mathcal{M} \quad \text{with} \quad \|x_L - x_g\|_2 < \|x_H - x_g\|_2,$$

at which it resumes motion along $\mathcal{M}$. This leave-point rule eliminates the full-boundary traversal that Bug1 incurs at each obstacle, but the method still relies only on local geometric collision information and provides no mechanism for deciding whether to displace an obstacle.

C. D* Lite

D* Lite [7] is an incremental heuristic search planner for navigation in partially known environments. Let $G_t = (V, E, c_t)$ denote a discretization of Ω into a uniform grid with edge cost $c_t : E \to \mathbb{R}_{\geq 0} \cup \{\infty\}$, where edges adjacent to known obstacles in $\mathcal{O}_t$ carry cost ∞ and remaining edges carry their geometric length. D* Lite maintains, for each vertex

$s \in V$, a goal-distance estimate $g(s)$ and a one-step lookahead value

$$\mathrm{rhs}(s) = \begin{cases} 0, & s = x_g, \\ \min_{s' \in \mathrm{Succ}(s)} \big(c_t(s, s') + g(s')\big), & \text{otherwise.} \end{cases}$$

Vertices with $g(s) \neq \mathrm{rhs}(s)$ are placed on a priority queue keyed by

$$k(s) = \big[\min(g(s), \mathrm{rhs}(s)) + h(x_t, s) + k_m, \\ \min(g(s), \mathrm{rhs}(s))\big].$$

where h is an admissible heuristic and k_m is a key offset that compensates for robot motion between replanning calls. The robot moves greedily by selecting

$$x_{t+1} \in \arg\min_{s' \in \mathrm{Succ}(x_t)} \big(c_t(x_t, s') + g(s')\big),$$

and only the affected vertices are updated when newly observed obstacles change c_t. D* Lite serves as the strongest classical search-based baseline because it explicitly replans as the known environment changes. It still treats obstacles as binary geometric constraints, so it cannot represent contact desirability or interaction risk.

D. Contact-Aware Baseline

The contact-aware baseline shares BEACON's local action-selection structure but replaces the belief-weighted interaction term with a fixed-weight surrogate. At each step, it selects

$$q_t^\star = \arg\min_{q \in Q_t} \Big[W_P J_{\mathrm{pos}}(q) + W_R J_{\mathrm{risk}}^{\mathrm{fixed}}(q) \\ + W_B J_{\mathrm{res}}(q)\Big],$$

where

$$J_{\mathrm{risk}}^{\mathrm{fixed}}(q) = \alpha\, J_{\mathrm{geo}}(q) + \beta\, J_{\mathrm{int}}^{\mathrm{fixed}}(q), \quad \alpha + \beta = 1,$$

and the interaction term uses static per-class costs:

$$J_{\mathrm{int}}^{\mathrm{fixed}}(q) = \min\left(1, \frac{\bar{c}\big(\hat{c}_{i^\star(q)}\big)}{c_{\max}}\right),$$

with $\hat{c}_{i^\star(q)}$ the most-likely interaction class for the nearest observed obstacle and $\bar{c}(\cdot)$ a fixed cost lookup. The weights $(\alpha, \beta, W_P, W_R, W_B)$ are constant across episodes. This baseline isolates the contribution of belief-aware reasoning from contact-awareness alone: any performance gap between BEACON and this baseline reflects the value of updating interaction estimates online rather than the value of permitting contact in the first place.

APPENDIX B
EVALUATION METRICS

Let N denote the total number of episodes. For episode i, let s_i denote the number of executed steps, $x_t^{(i)} \in \mathbb{R}^2$ the robot position at step t, c_i the number of logged contact events, and m_i the number of sensed obstacles.

Success rate is the fraction of episodes reaching the goal:

$$\mathrm{SuccessRate} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[\text{episode } i \text{ reaches the goal}].$$

Average steps is $\mathrm{AvgSteps} = \frac{1}{N} \sum_{i=1}^{N} s_i$.

Path length for episode i is the cumulative Euclidean distance $L_i = \sum_{t=1}^{T_i - 1} \|x_{t+1}^{(i)} - x_t^{(i)}\|_2$, where T_i is the number of recorded states. We report $\mathrm{AvgPath} = \frac{1}{N} \sum_{i=1}^{N} L_i$.

Average contacts is $\mathrm{AvgContacts} = \frac{1}{N} \sum_{i=1}^{N} c_i$, measuring how often the trajectory involved explicit obstacle contact.

Average sensed obstacles is $\mathrm{AvgSensed} = \frac{1}{N} \sum_{i=1}^{N} m_i$, reflecting the amount of local environment information revealed during execution.

Total energy. For each episode we also record the total energy expended,

$$\bar{E}_i = \sum_{t=0}^{T_i - 1} \big[\kappa_d \|x_{t+1}^{(i)} - x_t^{(i)}\|_2 + \mathbf{1}_{\mathrm{push}}^{(i,t)}\, E_{\mathrm{push}}^{(i,t)}\big],$$

combining a distance-proportional motion cost (with constant κ_d) and the contact energy from the force-energy model whenever the robot executes a push at step t. We report $\mathrm{AvgEnergy} = \frac{1}{N} \sum_{i=1}^{N} \bar{E}_i$.

APPENDIX C
EXPANDED RELATED WORK

Geometric and incremental planners. Classical motion planning treats obstacles as binary geometric constraints, where contact constitutes failure [3]. Bug-style algorithms [6] and incremental search methods such as D* Lite [7] reason entirely over geometry. They perform well when free space is abundant but provide no mechanism for deciding whether a blocking obstacle should be displaced or avoided. Sampling-based planners including Kinodynamic RRT* [5] and manifold-sample methods [4] extend feasibility to higher-dimensional dynamics but retain the collision-free assumption.

Contact-aware navigation. A growing body of work relaxes the collision-free assumption. IMPACT [1] uses vision-language models to score contact tolerance per obstacle, constructing a directional cost map for an A* planner that permits semantically acceptable contact in cluttered scenes. The Bench-Push benchmark [8] establishes pushing-based navigation as a practically important regime and provides standardized environments and metrics. Terrain-aware planners for legged systems assign traversal costs from fixed semantic labels [9], and semantic obstacle trackers derive class-dependent safety margins from segmented detections [2]. These approaches share a common pattern: semantic information is extracted at observation time and treated as fixed thereafter. The push-versus-avoid decision is resolved using cost weights set offline rather than updated during execution. BEACON addresses this gap by maintaining belief over interaction class and updating it as contact evidence accumulates.

Belief-space and uncertainty-aware planning. Belief-space methods including FIRM [10] and SLAP [11] track uncertainty over robot pose using sampling-based methods in belief space, primarily for localization-aware navigation. Partially observable task-and-motion planning [12] extends belief reasoning to object interactions and information gathering, primarily in tabletop manipulation settings. These methods

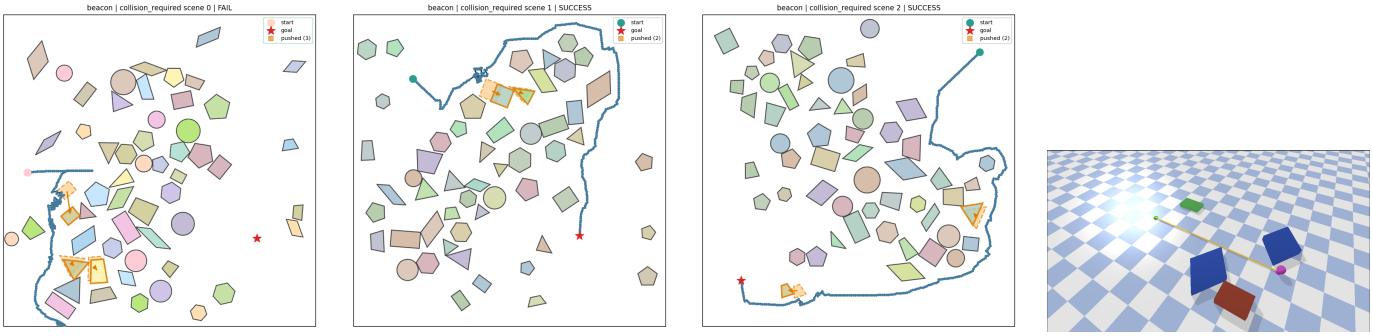
establish that belief reasoning is a productive substrate for planning under uncertainty, but they target a different regime than contact-rich 2D and 3D mobile navigation. The relevant uncertainty in our setting is over interaction class rather than pose, and the action space is local reactive selection rather than long-horizon TAMP.

Online replanning under semantic uncertainty. The Blindfolded Traveler's Problem [15] shows that contact is an informative signal for revising world models during execution, establishing the principle that observations made during navigation should propagate back to the planning layer. Context-Aware Replanning [13] triggers replanning when semantic map confidence falls below a threshold, using multi-view re-observation to correct labeling errors online. Taheri and Aksaray [14] handle unknown semantic label distributions via automata-theoretic formulations with online value iteration for temporal logic specifications. BEACON shares the underlying principle that execution evidence should update planning, but operationalizes it through belief-weighted local action scoring rather than confidence-triggered replanning or temporal logic synthesis. The integration of belief updates with reactive local action selection in a contact-rich navigation setting is, to our knowledge, what has not been studied empirically before.

APPENDIX D
QUALITATIVE COMPARISONS

Figure 1 provides representative qualitative results for BEACON in both 2D and 3D environments. The 2D panels show collision-required scenes where the robot must reason about contact to make progress, while the 3D panel illustrates the corresponding PyBullet visualization used for qualitative inspection.

Videos showing 3D movement are available at https://drive.google.com/drive/folders/1dd_IqZa2s9XZXGJ9uL2wfNO_24gfVwXI?usp=sharing.



(a) 2D scene 000 (b) 2D scene 001 (c) 2D scene 002 (d) 3D visualization

Fig. 1: Qualitative BEACON visualizations. The first three panels show 2D collision-required scenes, and the final panel shows the 3D PyBullet visualization.