Proceedings of the Twelfth International
Symposium on Combinatorial Search (SoCS 2019)

Intuitive, Reliable Plans with Contingencies: Planning with Safety Nets for Landmark-Based Routing

Kalyan Vasudev Alwala
Robotics Institute
Carnegie Mellon University

Margarita Safonova
National Robotics Engineering Center
Carnegie Mellon University

Oren Salzman
Robotics Institute
Carnegie Mellon University

Maxim Likhachev
Robotics Institute
Carnegie Mellon University

Abstract

We are interested in the problem of providing intuitive instructions for human agents to enable reliable navigation in unknown environments. Since the advent of GPS and digital maps, a common approach is to visually provide a planned path on a digital map defined in terms of actions to take at specific junctions. However, this approach relies on the agent to constantly and accurately localize itself. Furthermore, it comes in stark contrast to the way humans provide instructions—by leveraging known landmarks in the environment to both augment the description of the planned path as well as to allow to detect when the agent deviated from the planned path. Hence, there is need for assurable means of localization, an intuitive way of compactly conveying directions to agents and a systematic approach to account for human errors. To this end, our key insight is to employ known *landmarks* in the environment to overcome these challenges. We formally model this intuitive way to use landmarks for conveying instructions and for creating contingency plans. We present experiments demonstrating the efficacy of our approach both on synthetic environments as well as on real-world maps, computed using a smart-phone iOS application that we developed.

1 Introduction

We consider the problem of a human agent navigating in an unfamiliar environment from a given start to target location. Specifically, we are interested in the case where we are given a map of the environment and we want to account for human-navigation errors. One approach, enabled by technological advancements such as portable hand-held digital maps and Global Positioning System (GPS), is to follow a pre-computed path using navigation services (see, e.g., (iOS Maps contributors 2017; Google Maps contributors 2017)). These services both plan a shortest path (given some optimization criteria) as well as interactively show the agent's location on the map. However, these services require either that (i) the GPS-data is both accurate as well as updated in real-time and the agent constantly monitors its position or that (ii) the agent is able to accurately interpret and constantly localize itself on the map. When these conditions do not hold the agent may take wrong actions such as missing a turn or taking a turn at the wrong location. Hence, there is need for assurable means of localization, an intuitive way of compactly conveying directions to agents and a systematic approach to account for human errors. To this end, our key insight is to employ known *landmarks* in the environment to overcome these challenges.

Landmarks are features in the environment that are easily seen or recognized. They are commonly used in our everyday life both for localization and to provide directions, acting as triggers dictating when an agent should perform specific actions. For example, a common instruction given by humans may be in the form (see also Fig. 1)

> "... continue straight until landmark ℓ_0 (a gas station for example) and then turn right at the junction until the destination is reached. If you reached landmark ℓ_1 (a pizza store for example), you missed the turn and should immediately turn right at the following junction and then continue until the destination is reached."

Our main contribution is formally modeling this intuitive way to use landmarks for conveying instructions and for creating contingency plans.

After briefly describing the related work in Sec. 2, we start modeling our problem in Sec. 3 by considering the simple case when the agent always detects landmarks and there is no need to create contingency plans. In this case, the problem can be cast as a graph-planning problem which we refer to as *landmark-based routing*, or LBR. Computing minimal-cost paths (to be formally defined in Sec. 3) for LBR corresponds to computing shortest path on a graph.

We then continue in Sec. 4 to consider the event that landmarks may be missed and how to generate plans that are robust to such events. Roughly speaking, these are no longer plans but *policies*. Since we wish to account for every landmark that could be missed, computation of such policies is intractable as the number of landmarks increases. Moreover, this is not how people convey directions which would hinder the applicability of the proposed model. People typically give directions for how to detect a missed landmark or make sure that a following landmark will not be missed but they do not specify a full policy. Thus, we introduce the additional assumptions that the agent cannot miss two

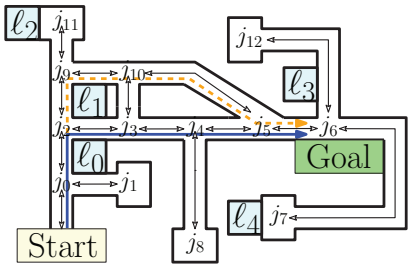


Figure 1: Landmark-based routing with safety nets. The solid blue path is the planned path to the goal from the start state. The dashed orange path is a safety-net plan that the agent would follow if landmark ℓ_0 was missed.

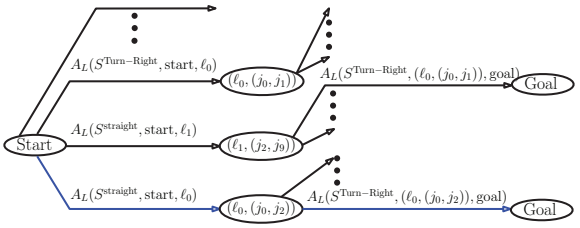


Figure 2: Partial depiction of G_{N} for the map depicted in Fig. 1. The minimal-cost path is depicted in blue.

consecutive landmarks and that it can miss at most n landmarks in total (for some given n). We call this problem *LBR with safety nets of degree* n. These assumptions allow us to cast the problem as a stochastic shortest-path problem while leveraging landmarks in the environment to guide the agent.

To efficiently solve this problem, we use PPCP (Likhachev and Stentz 2009). PPCP was shown to outperform alternative algorithms but requires that specific assumptions on the problem hold. While our model does not directly satisfy these assumptions (Sec. 5) we experimentally show in Sec. 6 that PPCP produces solutions much faster than its counterparts without compromising much on the solution quality. In our evaluation, presented in Sec. 6, we solve the problem of planning with safety nets for both synthetic as well as for real environments using a smart-phone iOS application that we developed. We demonstrate the type of paths produced as well as the favorable traits of PPCP when compared to alternative approaches. We conclude in Sec. 7 with a discussion on the limitations of our models and directions for future work.

2 Related work

2.1 Landmarks in human-agent interfaces

Since the advent of GPS and digital maps, there has been growing interest in using landmarks to augment interfaces for planning and for providing instructions to human agents (Beeharee and Steed 2006; Hile et al. 2008; 2009; Wither et al. 2013).

The aforementioned papers consider augmenting landmark images to improve the spatial understanding of the route followed by the user. However, all of these approaches use landmarks as a post-processing step to the already-computed routes. Furthermore, there is no clear structure on how and when to add landmarks to the routes nor do they address the problem of uncertainty in landmark detection.

2.2 Robot planning using landmarks

Over the last two decades, there has been extensive work on planning with landmarks, particularly in the context of landmark-based robot navigation. Early work can be traced to Latombe et al. (Lazanas and Latombe 1995a; Latombe 1990; Lazanas and Latombe 1995b) who considered the problem of planning under uncertainty where landmarks are considered as zones with perfect localization. Outdoor vehicle navigation is another domain in which landmarks have been used to aid in the localization of the navigating vehicles (Hait, Siméon, and Taïx 1999; Fraichard and Mermond 1998; Gonzalez and Stentz 2007). Landmark-based navigation has also been used in the context of multiple agents traversing unknown environment (Busquets, Sierra, and de Màntaras 2003). Finally, there has been some recent work on learning landmark representations and using them to visually aid robot navigation in unknown environments (Gupta et al. 2017).

2.3 Planning for MDPs

Our problem of landmark-based routing with Safety Nets of degree n falls under the specific category of planning with sensing which, in our formulation, is a Markov Decision Process (MDP)[1]. There have been multiple algorithm proposed to solve MDPs such as RTDP (Bonet 1998), which computes optimal policies using dynamic programming in the belief MDP representation. As we show in our experiments (Sec. 6), in our domain RTDP fails to scale well in large environments because of its need to perform an update step on every single state in the state space. LAO* (Hansen and Zilberstein 2001) is another algorithm that can be used for planning on MDPs. Although it leverages advantages of heuristic search to a certain extent, this approach still is bottlenecked by a dynamic-programming step and in our domain proves (empirically) to be too slow. The approach we adopt in our work is PPCP, a heuristic-search algorithm which was shown to perform better, in terms of scalability and computation time, the alternative algorithms mentioned (Likhachev and Stentz 2009).

Our work is closely related to offline contingent planning (Domshlak 2013) where an agent is allowed to misinterpret a finite number of actions. Similarly, it bears resemblance to fully observable non-deterministic (FOND) planning (Muise, McIlraith, and Beck 2012; Muise, Belle, and McIlraith 2014) for finding goal-oriented plans. This work was recently extended to the setting of maximizing an agent's probability of reaching a goal while preserving guarantees on the computed plans (Camacho et al. 2015).

[1] In its general form, planning with sensing is a Partially Observable Markov Decision Process (POMDP). It certain cases, including ours, it can be modeled as an MDP (see assumptions in Sec. 4).

3 Landmark-based Routing (LBR)

In this section, we formulate and solve the problem of landmark-based routing under the assumption that there is no uncertainty in landmark detection. This can be seen as a warm-up for Sec. 4 where we generalize this framework to handle the case of uncertainty in landmark detection.

Informally, given a roadmap and a set of landmarks, the objective of landmark-based routing (LBR) is to generate plans specified in terms of *actions* to be taken when specified *landmarks* are observed. This can be done while optimizing for a variety of criteria such as length of the path, number of turns on the path, number of landmarks used on the path etc.

We start by formally defining the notions of a roadmap and landmarks (Sec. 3.1). We then describe how these are used to define actions (Sec. 3.2) and then describe how these are used to formulate and solve the problem of LBR (Sec. 3.3).

3.1 Road networks and landmarks

Given a map of the environment, a road network $G_{\mathrm{N}} = (V_{\mathrm{N}}, E_{\mathrm{N}})$ is a planar directed graph generated using this map. Here, each vertex $v \in V_{\mathrm{N}}$ is associated with the location of a junction on the map and each edge $e = (u, v) \in E_{\mathrm{N}}$ corresponds to a road connecting two junctions. We assume that there is a function $W_{\mathrm{N}} : V_{\mathrm{N}} \times V_{\mathrm{N}} \to \mathbb{R}$ defining a cost of reaching one vertex from another in the road network.

A set of landmarks $L = \{l_1, \ldots, l_k\}$ in the environment corresponds to k distinct locations in the map. We assume that each landmark $l \in L$ is associated with a finite number of edges, each corresponding to a road from which the landmark can be seen. Namely, there exists some mapping $M : L \to 2^{E_{\mathrm{N}}}$. For instance, for the road network example introduced in Fig. 1, $M(\ell_0) = \{(j_0, j_1), (j_1, j_0), (j_0, j_2), (j_2, j_0), (j_2, j_3), (j_3, j_2)\}$. Finally, we assume that the cost function W_{N} can be extended to take landmarks into account. Namely, that $W_{\mathrm{N}} : (V_{\mathrm{N}} \cup L) \times (V_{\mathrm{N}} \cup L) \to \mathbb{R}$.

3.2 Junction actions and landmark actions

Landmark-based routing, or LBR is a method for specifying directions, or routes, in terms of *actions* to be taken when designated landmarks are observed. This allows to generate intuitive-to-follow directions for human agents. These directions, which are constructed of an alternating sequence of landmarks and junction actions, alleviate the burden of providing (and receiving) instructions in terms of every road and junction passed along the route. We start by defining the notion of a *junction action* which is a method to specify which outgoing road to take when approaching a junction from an incoming road. We then continue to define the notion of a *junction-action sequence* which, in turn, will be used to define *landmark actions*.

Junction actions A junction action $A_{\mathrm{J}} : (V_{\mathrm{N}}, E_{\mathrm{N}}) \to E_{\mathrm{N}}$ specifies which outgoing road to take when approaching a junction from an incoming road. Let $v \in V_{\mathrm{N}}$ be a junction and let $e_{\text{in}}, e_{\text{out}} \in E_{\mathrm{N}}$ be incoming and outgoing edges from v, respectively. In this work we consider two specific types of junction actions $A_{\mathrm{J}}^{\text{Straight}}$ and $A_{\mathrm{J}}^{\text{Turn X}}$.

The first action $A_{\mathrm{J}}^{\text{Straight}}$ is used to specify a "natural" successor (e.g., "stay on the road") of an incoming edge to a junction. Namely, $A_{\mathrm{J}}^{\text{Straight}}(v, e_{\text{in}}) = e_{\text{out}}$ if e_{out} is the "natural" route an agent would take when approaching v from e_{in}. We assume that this function is well defined for any $v \in V_{\mathrm{N}}$.[2]

The second action $A_{\mathrm{J}}^{\text{Turn-X}}$ is used to specify an action that involves taking an outgoing edge that is *not* represented by $A_{\mathrm{J}}^{\text{Straight}}$. For example, $A_{J}^{\text{Turn-2nd-Right}}(v, e_{\text{in}}) = e_{\text{out}}$ if e_{out} is reached by taking the second right-hand turn at junction v when coming from e_{in}.

Landmark vertices To uniquely define the location of the agent with respect to a given landmark, we will define the notion of a *landmark vertex* which is a pair (l, e) such that $l \in L$ and $e \in M(l)$. Namely, (l, e) represents a state in which landmark l is observed by the agent while on the road segment e. Let $V_{\mathrm{L}} = \{(l, e) | l \in L \text{ and } e \in M(l)\}$ be the set of all landmark vertices. Landmark vertices, together with junction actions, will be used to define landmark actions—the method we use to convey instructions to the agent.

Landmark actions A landmark action is a method of conveying a route between two landmark vertices. It is defined as the act of executing junction actions at every junction encountered on the path until the designated landmark is observed.

A *junction-action sequence* $S = \{A_{\mathrm{J}}^{\mathrm{a}1}, A_{\mathrm{J}}^{\mathrm{a}2}, \ldots\}$ is a predefined (infinite) sequence of junction actions. Let $\mathcal{S}$ denote the set of all junction-action sequences considered.

Given some landmark vertex $v_0 = (l_0, e_0)$, a junction-action sequence $S \in \mathcal{S}$ induces a sequence of landmark vertices $V(v_0, S) = \{v_1, v_2, \ldots\}$ where $v_i = (l_i, e_i)$ is the i'th landmark that will be visited when executing the junction-action sequence S on the road network G_{N}. Note that $V(v_0, S)$ could (i) be a finite sequence and (ii) consist of repeated landmark states.

Here after, we only consider a few special sequences in $\mathcal{S}$ to keep the instructions provided to the agent simple and intuitive. Specifically, we consider the following junction-action sequences:

- $S^{\text{Straight}} := \{A_{\mathrm{J}}^{\text{Straight}}, A_{\mathrm{J}}^{\text{Straight}}, \ldots\}$. Namely, the agent executes the junction action $A_{\mathrm{J}}^{\text{Straight}}$ at every junction encountered.
- $S^{\text{Turn-X}} := \{A_{\mathrm{J}}^{\text{Turn-X}}, A_{\mathrm{J}}^{\text{Straight}}, A_{\mathrm{J}}^{\text{Straight}}, \ldots\}$. Namely, the agent executes the junction action $A_{\mathrm{J}}^{\text{Turn-X}}$ at the first junction followed by $A_{\mathrm{J}}^{\text{Straight}}$ at the remainder of the junctions encountered.

Given some landmark vertex $v_0 = (l_0, e_0)$ and a junction-action sequence $S \in \mathcal{S}$, a *landmark action* $A_L : (S, V_L, L) \to V_L$ is a mapping instructing the agent to execute the junction-action sequence S from its current position $v_0 \in V_L$ until some landmark $l' \in L$ is reached. Note that a landmark action is only defined if there exists some

[2] In certain junctions, such as T-junctions, the notion of "straight" is not well defined. However, we assume that the ambiguity with regards to a "natural" action is solved by, say, assuming that this is always a right-hand turn.

edge $e' \in M(l')$ such that $(l', e') \in V(v_0, S)$. In such cases we have that $A_L(S, v_0, l') = (l', e')$.

3.3 LBR problem formulation

We are now ready to describe our formulation of LBR. Given the list of landmarks L and the directed road-network G_{N}, the problem of LBR can be cast as a shortest-path problem on a new graph $G_{\text{L}} = (V_{\text{L}}, E_{\text{L}})$ which we call a *landmark graph*.

The vertices V_{L} of G_{L} are the set of all landmark vertices. Namely,

$$V_{\text{L}} = \{(l, e) \mid l \in L \text{ and } e \in M(l)\}.$$

An edge exists in G_{L} between two landmark vertices $v_0 = (\ell_0, e_0)$ and $v_1 = (\ell_1, e_1)$ if there exists some junction-action sequence S and some landmark action A_L such that $A_L(S, (\ell_0, e_0), \ell_1) = (\ell_1, e_1)$. In other words, road segment e, from which l can be seen, will be reached by executing the sequence S of junction-actions. The cost of this edge is the sum of costs in G_{N} of the edges traversed when following the landmark action $A_L(S, (\ell_0, e_0), \ell_1)$.

Thus, given the start vertex $v_s \in V_{\text{L}}$, goal state $v_g \in V_{\text{L}}$ and the landmark graph G_{L}, LBR with no uncertainty in landmark detection corresponds to computing the shortest path in G_{L}. This problem can be solved using any search algorithm such as A* (Hart, Nilsson, and Raphael 1968). See Fig. 2.

4 LBR with safety nets

In Sec. 3 we formulated the problem of LBR under the assumption that every landmark is detected. In this section we relax this assumption to allow for some uncertainty in landmark detection (following our running example, this corresponds to the case where the agent misses the gas station). This is done by introducing the notion of *safety nets*. The objective of LBR with safety nets is to compute a plan that minimizes the *expected* cost to the goal. To this end, we make the following assumptions,

A1 Uncertainty is only in detecting landmarks (and not in detecting junctions).

A2 The start state is fully known (i.e., no uncertainty), and similarly, there is no uncertainty in detecting a goal state.

A3 The likelihood of detecting a landmark is independent of whether the agent previously observed that landmark.

A4 The agent may miss at most n landmarks.

A5 When executing any particular action (i.e., following a single instruction in the plan), the agent may miss at most one landmark (i.e., it won't miss two consecutive landmarks).

In assumption **A4**, n, referred to as the *safety net degree*, is a user-defined parameter. After missing n landmarks, the agent is assumed to detect all landmarks remaining in the plan. Not only does this assumption make plans look more intuitive to human agents, but it also makes planning more scalable as we show in our experimental analysis.

4.1 Safe landmark actions

In this section we use **A1**-**A5** to extend the notion of landmark actions. We start by extending the notion of landmark vertices to *safety-net augmented landmark vertices* (or sn-augmented landmark vertices) that account for the number of landmarks that the agent missed. We then use these to extend the notion of landmark actions to *safety-net augmented landmark actions* (or sn-augmented landmark actions) which account for the event that the agent misses a landmark.

SN-augmented landmark vertices An *sn-augmented landmark vertex* (v, ρ) uniquely defines the agent's state. Here, $v \in V_{\text{L}}$ denotes the agent's location and $\rho \in [n]$ denotes the total number of landmarks that the agent is still allowed to miss (the remaining safety net). Let $X_{\text{Safe}} = V_{\text{L}} \times [n]$ denote the set of all such safe landmark vertices.

Given an sn-augmented landmark vertex $x = (v, \rho)$, we will denote by $v(x)$ and $\rho(x)$ the landmark state v and the remaining number of landmarks that the agent is allowed to miss, respectively.

SN-augmented landmark actions An *sn-augmented landmark action* $A_{\text{Safe}} : (\mathcal{S}, X_{\text{Safe}}, 2^L) \to 2^{X_{\text{Safe}}}$ is a method of conveying a single instruction while accounting for uncertainty in detecting a landmark. To this end, this instruction includes the primary landmark that the agent is supposed to be looking for, as well as a backup landmark that is used to detect the case of missing the primary landmark. An example of such an instruction would be "follow this street until you see a gas station; however, if you see a a pizza store then you missed the gas station." In this example, the primary landmark is gas station and the backup landmark is the pizza store.

Note that **A5** ensures that the backup landmark will be detected. Furthermore, from **A4**, it follows that once n landmarks have been missed, there is no need to provide a backup landmark. Similarly, **A2** ensures that when reaching the goal there is no need to provide a backup landmark.

In the following we formalize these cases. Given a state $x \in X_{\text{Safe}}$ and a junction-action sequence $S \in \mathcal{S}$, we define the following types of sn-augmented landmark actions.

In case both a primary landmark ℓ_1 and a backup landmark ℓ_2 are utilized, the (non-deterministic) outcomes of the sn-augmented landmark action are given by:

$$A_{\text{Safe}}(S, x, \{\ell_1, \ell_2\}) = \{(v_1, \rho(x)), (v_2, \rho(x) - 1)\} \quad (1)$$

where $\rho(x) > 0$, $v_1 = A_L(S, v(x), \ell_1)$, $v_2 = A_L(S, v(x), \ell_2)$ and v_1 appears before v_2 in the ordered sequence $V(v(x), S)$ for some $v_1, v_2 \in V_L$.

In case n landmarks have already been missed, the sn-augmented landmark action is defined as:

$$A_{\text{Safe}}(S, x, \{l_1\}) = \{(v_1, 0)\} \quad (2)$$

where $\rho(x) = 0$ and $v_1 = A_L(S, v(x), \ell_1)$ for some $v_1, \in V_L$.

Finally, if $v_g = (\ell_g, e_g) \in V_{\text{L}}$ is the goal landmark state, the sn-augmented landmark action is defined as:

$$A_{\text{Safe}}(S, x, \{\ell_g\}) = \{(v_g, \rho(x))\} \quad (3)$$

where $v_g = A_L(S, v(x), \ell_g)$.

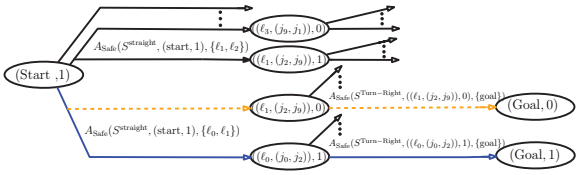


Figure 3: The plan with degree of safety net ($n = 1$) for the scenario introduced in Fig. 1.

Transition probabilities SN-augmented landmark actions defined in Eq. 1 are associated with two possible outcomes, corresponding to detecting and missing the primary landmark. For each sn-augmented landmark action $A_{\text{Safe}}(S, x, \ell_1, \ell_2)$ there are associated probabilities p_1 and p_2 (where $p_2 = 1 - p_1$). Here, p_1 and p_2 correspond to the probability to detect and miss landmark ℓ_1 when taking the junction-action sequence S from state x, respectively. The sn-augmented landmark actions defined in Eq. 2 and 3 are deterministic.

4.2 LBR with safety nets problem formulation

We are now ready to describe our formulation of LBR with safety nets. Given the list of landmarks L, the directed road-network G_{N}, start and goal landmark states $v_s, v_g \in V_{\text{L}}$ and a safety net degree n, the problem of LBR with safety nets can be cast as an *expected shortest-path* problem on a new graph $G_{\text{Safe}} = (X_{\text{Safe}}, E_{\text{Safe}})$ which we call a *safe landmark graph*.

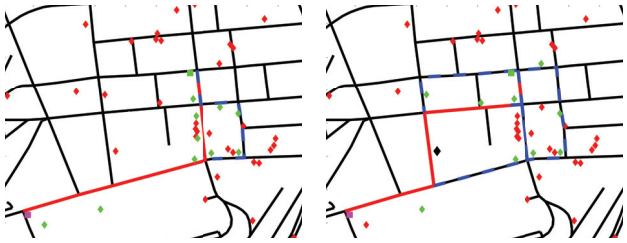
As mentioned in Sec. 4.1, the set of vertices of G_{Safe} is defined as $X_{\text{Safe}} = V_{\text{L}} \times [n]$. An edge exists in G_{Safe} between two sn-augmented landmark vertices $x_0 = (v_0, \rho_0)$ and $x_1 = (v_1, \rho_1)$, where $v_1 = (\ell_1, e_1)$, if there exists some junction action sequence S and some sn-augmented landmark action A_{Safe} such that $x_1 \in A_{\text{Safe}}(S, x_0, L')$ and $\ell_1 \in L'$. The cost of this edge is the sum of the costs in G_{N} of the edges traversed when following the landmark action $A_{\text{Safe}}(S, x_0, L')$. The probability of this edge is the probability associated with taking this action and reaching x_1.

We set $x_s = (v_s, n)$ and $X_g = \{(v_g, i) \mid i \in [n]\}$ to be the start state and the set of goal states, respectively. We define a *plan* as a tree on the graph G_{Safe} rooted at x_s where all the leaves belong to X_g, and set $\Pi(x_s, X_g)$ to be the set of all possible plans. The *cost* and *probability* to reach a leaf state $x_g \in X_G$ of a plan $\pi \in \Pi(x_s, X_g)$ is the sum and product of the costs and probabilities along the edges of π from x_s to X_g, respectively. The cost of a plan $\pi \in \Pi(x_s, X_g)$, denoted by $c(\pi)$, is the expected cost to reach a goal state in X_g. Let $\pi^* = \arg\min_{\pi \in \Pi(x_s, X_g)} c(\pi)$ denote the optimal plan and $c^* := c(\pi^*)$ denote its cost.

Finally, we note that a safety net is a plan; it is a sub tree of the original policy rooted at a backup landmark. An example illustrating a 1-degree safety net plan can be found in Fig. 3.

5 Efficient Planning with Safety Nets

The problem formulated in Sec. 4 is an MDP. It can be solved using a variety of existing approaches like RTDP, LAO* and PPCP. Among these methods, as we demonstrate in Sec. 6, PPCP scales the best on our domain while generating near-optimal solutions.



(a) (b)

Figure 4: Different 1-degree safety-net policies generated under different landmark-detection probabilities. Road network is marked in solid black, start and goal states are marked by magenta and green squares, respectively and landmarks used by the policy are marked using green and black diamonds for detection probabilities of 0.5 and 0.99, respectively. Finally, landmarks not used by the policy are marked using red diamonds and paths with 1-degree and zero-degree safety nets are marked in solid red and dashed blue, respectively.

5.1 Probabilistic Planning with Clear Preferences

PPCP is a heuristic search framework for computing policies, by running a series of deterministic A*-like searches. This approach is often much faster than alternative approaches but provides guarantees on solution optimality only when certain assumptions are true (Likhachev and Stentz 2008). In particular, it requires that *clear preferences* on multiple outcomes of stochastic actions, as described below, are known and given to the planner.

Clear preferences Under the clear preferences assumption (Likhachev and Stentz 2009; Neuman and Likhachev 2013), for any action with more than one outcome, we are given an outcome which is "clearly preferred" over the other outcomes. Specifically, for any state x and action a, the clearly-preferred outcome x' satisfies the following condition,

$$x' = \arg \min_{y \in succ(x,a)} c(x, a, y) + c^*(y) \qquad (4)$$

where, $succ(x, a)$ is the set of all the successors of x generated by executing action a, $c(x, a, y)$ is the cost of executing action a at state x and ending up at state y, and $c^*(y)$ is the expected cost of executing an optimal policy from the outcome state y.

Intuitively, the clear-preference assumption in our domain can be interpreted as "it is always better for the agent to be able to detect the primary landmark than to miss it and detect the backup landmark instead." More formally, given any state $x \in X_{\text{Safe}}$ and its successors x_1, x_2 for some sn-augmented landmark action $a_s = A_{\text{Safe}}(s, x, \{l_1, l_2\})$ where $x_1 = (v_1, \rho(x))$ and $x_2 = (v_2, \rho(x) - 1)$, the state x_1 is a clearly preferred state. Unfortunately, in our setting the clear preferences assumption does not hold. For it to hold,

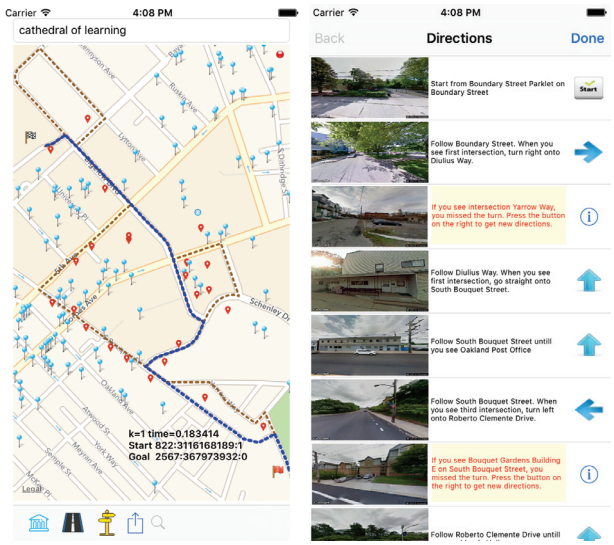


Figure 5: An illustration of our iOS application. *Left:* The screenshot displays the entire plan with safety net degree $n = 1$. The start and goal states are marked using red and checkered flags, respectively. All the landmarks are highlighted using blue circular markers. Paths with 1-degree and zero-degree safety nets are marked in solid blue and dashed brown, respectively. Landmarks marked in red are the ones used on the plan. *Right:* The semantic instructions given to the agent. The primary and backup instructions are marked in black and red, respectively.

the expected cost would have to satisfy that $c(x, a_s, x_1) + c^*(x_1) \leq c(x, a_s, x_2) + c^*(x_2)$ where $c(x, a_s, c_1)$ and $c(x, a_s, c_2)$ represent the cost of executing a_s at x and reaching x_1 and x_2 respectively. While intuitively it seems that detecting a primary landmark is always better than missing it, the reason why it may not be the case is that missing a primary landmark reduces the remaining safety net degree. This reduction may potentially allow the agent to have a lower cost plan since there is less chances left to miss a landmark.

While the violation of clear preferences precludes the guarantee of solution optimality, we demonstrate experimentally in the following section that solutions reported are near optimal.

6 Implementation and results

In this section we present experiments demonstrating our model's capabilities to capture the intuitive notion of landmark based routing both in simulation (Subsec. 6.1) as well as in real environments (Subsec. 6.2). We conclude with simulations comparing our PPCP-based implementation to alternative algorithms (Subsec. 6.3).

Our implementation of the framework is in C++. All our experiments were carried out on a machine with 8 GB RAM clocked at 2.7 Ghz, operating on Ubuntu 14.04 operating system. For algorithms that require a heuristic function, we used the Euclidean distance between the perpendicular projections of the landmarks in the current state and the start state onto their respective connected roads.

6.1 Model evaluation

Our first experiments demonstrate how policies should be biased towards taking paths that contain prominent landmarks (landmarks with high-detection probabilities). Consider Fig. 4 where we computed the optimal policy once when all the probabilities of detecting the landmarks are 0.5 (Fig. 4a) and once after changing one landmark to be prominent (Fig. 4b).

As can be seen, using the prominent landmarks requires additional contingency plans, or a safety net, but this will happen with small probability making this policy the optimal one.

6.2 Real-world environment

To demonstrate landmark-based routing for human-agent navigating, we developed a smart phone application (iOS application) that allows to generate intuitive plans. The application queries routes and landmark data from an on-line map database (OpenStreetMap contributors 2017) and generates plans using our PPCP-based planning framework. As shown in Fig. 5, the application generates instructions alongside with visuals of the landmarks that need to be tracked. These instructions are derived from the actions in our planning framework. The agent also has access to the entire plan marked on the map. Average planning time for finding policies in this setting for a safety net degree of $n = 1$ is 2.8 secs.

While extensive user studies using this application is beyond the scope of this work (see Sec. 7), anecdotal feedback is that it is intuitive to use.

6.3 Planning times on synthetic environments

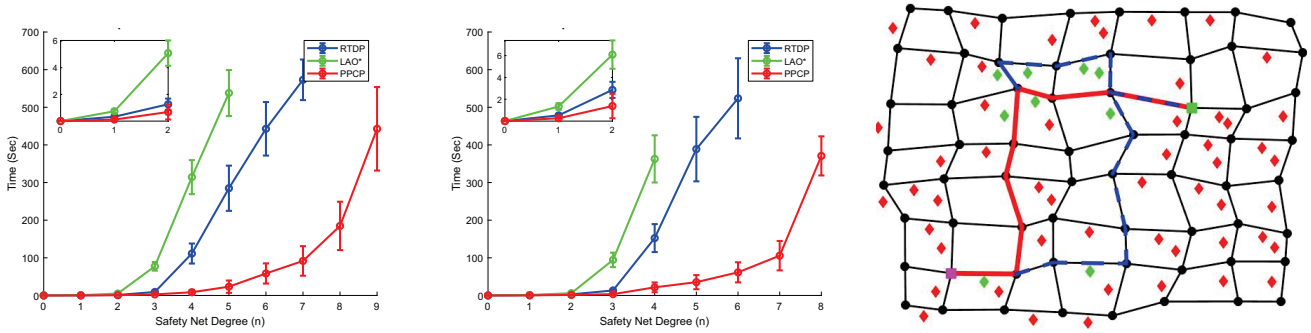
To evaluate the performance of our planner we generated a set of synthetic environments with randomly scattered landmarks (see Fig. 6c). We compared our PPCP-based planner to RTDP (Bonet 1998) and LAO* (Hansen and Zilberstein 2001), allowing each planner to run at most ten minutes.

For each environment, we considered 30 randomly-placed start and goal pairs. For each query, we considered a safety net degree ranging from zero to nine. Algorithm runtimes are depicted in Fig. 6a and 6b.

Finally, we note that (i) for all three planners, the memory consumption is roughly identical and that (ii) the quality of the solution obtained by PPCP was no more that 0.01% higher than the optimal cost computed by RTDP and LAO*. This demonstrates empirically that the significant speedup obtained by relaxing optimality guarantee and using PPCP comes with a very small compromise on the solution quality obtained by the planner.

7 Conclusion and future work

Our primary focus in this work was formally *modeling* the intuitive way in which landmarks are used for conveying instructions and for creating contingency plans by human users. We introduced the framework of safety nets and



(a) Results for 15×15 road network (b) Results for 25×25 road network (c) Simulated environment

Figure 6: Planning time comparison of PPCP, RTDP and LAO* for varying degrees of safety net on the simulated environments consisting of (a) 15×15 junctions and (b) 25×25 junctions respectively. (c) An illustration of a plan with safety net degree of one on one synthetic environment. Here, the road network and junctions are marked in black. All landmarks are marked using red diamond symbols, the start and goal state are marked in magenta and green squares, respectively. Paths with 1-degree and zero-degree safety nets are marked in solid red and dashed blue, respectively. Finally, all the landmarks considered on the plan are marked using green diamonds.

showed that the problem of planning with safety nets under the assumption that the agent is allowed only to miss at most n landmarks is a tractable problem. This assumption allows us to construct a graph-planning problem that can be efficiently solved using PPCP. Experimentally, it outperforms and scales better than popular alternatives while generating near-optimal solutions.

Immediate next steps are to perform user studies to (i) verify the validity of our model, specifically assumptions **A4** and **A5** and to (ii) verify the intuitiveness of the application developed. Assumption **A4** is bound to hold for large enough values of n and a natural question that we leave for future work is how to determine the value of n for a given environment (and possibly for a given user). Assumption **A5** may prove to be too restrictive for certain users. Using the framework we developed, we can easily relax it to assume that the user can miss at most k consecutive landmarks (for some small k). This will relax the assumptions that we take but will come at the price of more cumbersome instructions. An example of such an instruction for $k = 2$ would be "follow this street until you see a gas station; however, if you see a a pizza store then you missed the gas station. If you see a coffee shop, you missed both the gas station and the pizza store".

While the immediate applicability of our framework depends on the aforementioned user studies. Both the model we developed as well as the algorithmic tools to plan in such a model are general enough to accommodate for slight changes to our assumptions.

Finally, in future work, we aim to develop an extension to our algorithm that is capable of handling uncertainty in the start state. Addressing this would allow for the framework to be used to perform re-planning in case the agent gets lost but has some idea where he or she might be located.

8 Acknowledgments

We thank Apple for their support of this work.

References

Beeharee, A. K., and Steed, A. 2006. A natural wayfinding exploiting photos in pedestrian navigation systems. In *Human-computer interaction with mobile devices and services*, 81–88. ACM.

Bonet, B. 1998. Solving large POMDPs using real time dynamic programming. In *AAAI Fall Symposium on POMDPs*.

Busquets, D.; Sierra, C.; and de Màntaras, R. L. 2003. A multiagent approach to qualitative landmark-based navigation. *Autonomous Robots* 15(2):129–154.

Camacho, A.; Muise, C.; Ganeshen, A.; and McIlraith, S. 2015. From FOND to probabilistic planning: Guiding search for quality policies. In *Workshop on Heuristic Search and Domain Independent Planning, ICAPS*.

Domshlak, C. 2013. Fault tolerant planning: Complexity and compilation. In *ICAPS*, 64–72.

Fraichard, T., and Mermond, R. 1998. Integrating uncertainty and landmarks in path planning for car-like robots. In *IFAC Symp. on Intelligent Autonomous Vehicles*, volume 25, 27.

Gonzalez, J. P., and Stentz, A. 2007. Planning with uncertainty in position using high-resolution maps. In *ICRA*, 1015–1022.

Google Maps contributors. 2017. https://www.google.com/maps.

Gupta, S.; Fouhey, D.; Levine, S.; and Malik, J. 2017. Unifying map and landmark based representations for visual navigation. *arXiv:1712.08125*.

Hait, A.; Siméon, T.; and Taïx, M. 1999. Robust motion planning for rough terrain navigation. In *IROS*, volume 1, 11–16.

Hansen, E., and Zilberstein, S. 2001. LAO*: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence* 129(1):35–62.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* 4(2):100–107.

Hile, H.; Vedantham, R.; Cuellar, G.; Liu, A.; Gelfand, N.; Grzeszczuk, R.; and Borriello, G. 2008. Landmark-based pedestrian navigation from collections of geotagged photos. In *Mobile and ubiquitous multimedia*, 145–152. ACM.

Hile, H.; Grzeszczuk, R.; Liu, A.; Vedantham, R.; Košecka, J.; and Borriello, G. 2009. Landmark-based pedestrian navigation with enhanced spatial reasoning. In *Inter. Conf. on Pervasive Computing*, 59–76.

iOS Maps contributors. 2017. Retrieved from https://www.apple.com/ios/maps . https://www.apple.com/ios/maps.

Latombe, J.-C. 1990. Robot motion planning.

Lazanas, A., and Latombe, J.-C. 1995a. Landmark-based robot navigation. *Algorithmica* 13(5):472–501.

Lazanas, A., and Latombe, J.-C. 1995b. Motion planning with uncertainty: a landmark approach. *Artificial intelligence* 76(1):287–317.

Likhachev, M., and Stentz, A. 2008. PPCP: The proofs. *University of Pennsylvania, Philadelphia, PA," Tech. Rep*.

Likhachev, M., and Stentz, A. 2009. Probabilistic planning with clear preferences on missing information. *Artificial Intelligence* 173(5):696–721.

Muise, C.; Belle, V.; and McIlraith, S. 2014. Computing contingent plans via fully observable non-deterministic planning. In *AAAI*, 2322–2329.

Muise, C.; McIlraith, S.; and Beck, C. 2012. Improved non-deterministic planning by exploiting state relevance. In *ICAPS*.

Neuman, B., and Likhachev, M. 2013. Planning with approximate preferences and its application to disambiguating human intentions in navigation. In *ICRA*, 415–422.

OpenStreetMap contributors. 2017. http://www.openstreetmap.org.

Wither, J.; Au, C. E.; Rischpater, R.; and Grzeszczuk, R. 2013. Moving beyond the map: Automated landmark based pedestrian guidance using street level panoramas. In *Human-computer Interaction with Mobile Devices and Services*, 203–212.