Submitted, 2016 Conference on Decision and Control (CDC)
http://www.cds.caltech.edu/~murray/preprints/dlm16-cdc_s.pdf

Identifying and exploiting tolerance to unexpected jumps in synthesized strategies for GR(1) specifications

Sumanth Dathathri Scott C. Livingston Richard M. Murray

***Abstract*— When used as part of a hybrid controller, finite-memory strategies synthesized from LTL specifications rely on an accurate dynamics model in order to ensure correctness of trajectories. In the presence of uncertainty about this underlying model, there may exist unexpected trajectories that manifest as unexpected transitions under control of the strategy. While some disturbances can be captured by augmenting the dynamics model, such approaches may be conservative in that bisimulations may fail to exist for which strategies can be synthesized. In this paper, we characterize the tolerance of such hybrid controllers - synthesized for generalized reactivity(1) specifications- to disturbances that appear as unexpected jumps (transitions) to states in the discrete strategy part of the controller. As a first step, we show robustness to certain unexpected transitions that occur in a finite-manner, i.e., despite a certain number of unexpected jumps, the sequence of states obtained will still meet a stricter specification and hence the original specification. Additionally, we propose algorithms to improve robustness by increasing tolerance to additional disturbances. A robot gridworld example is presented to demonstrate the application of the developed ideas and also to obtain empirical computational and memory cost estimates.**

I. Introduction

The ability of strategies synthesized from formal specifications to be tolerant to unexpected perturbations (disturbances or uncertainty or unexpected failures) is important - more so for safety-critical applications. This is an area of concern with reactive strategies because they are not error-resilent. Even with disturbances that are not critical to the system, but were not accurately modeled during synthesis, no guarantees can be provided about satisfaction of the temporal-formula used for synthesis. Though sometimes these uncertainties can be modelled through the dynamics, it may be the case that it is not possible to synthesize a winning strategy with the uncertainty.

After a disturbance, if resynthesis is done from the perturbed point, there are no current results that provide guarantees about the execution with segments from two separate strategies. In this paper, we make progress towards enhancing the tolerance of strategies synthesized to satisfy specifications in the generalized reactivity(1) (GR(1)) fragment of linear-temporal logic (LTL) [12], [13]. GR(1) formulae are considered because they are quite expressive in terms of temporal properties captured, yet symbolic synthesis algorithms are possible if relatively low computational complexity [3], [7], [11]. The first result we show is that by trivially refining a strategy synthesized to satisfy a GR(1) formula, a strategy that is robust to certain unexpected perturbations and guarantees i.e winning against a stricter formula can be generated. Then, exploting this tolerance, we propose multiple algorithms that combine separately synthesized strategies to form a single robust winning strategy. It is often desired that the system can recover from these glitches (uncertainities/noise) and function normally and this be done without resynthesizing the entire strategy again. It is also desirable that the strategies allow for recovery from faults whenever possible. In this regard, we propose one such approach which lets us recover from glitches without a complete resynthesis.

Understanding the behavior of systems to disturbances and uncertainties has been extensively studied in control theory and more recently, for reactive controllers and their synthesis. In [10], [15], [14], the robustness considered is in terms of bounded input-output deviation. This relates directly to the prevalent notion in control for robustness[17], where controllers are designed to ensure bounded disturbances lead to bounded deviations from nominal-behavior for the system. In this work, the tolerance to disturbances is in the form of satisfaction of a formula representing the desired system behavior. However, we do not yet propose a measure on this interpretation of $robustness$. In [2], the effect of disturbances on system behavior is quantified. The focus here is to synthesize robust systems that degrade gracefully - smallest number of system failures possible but not primarily directed on GR(1) specifications. Some existing work on notions of robustness in terms of satisfaction or violation of a formula can be found in [16], [4], [1]. The main objective in our paper is to understand and augment the robustness of pre-existing strategies for recovery from disturbances in constrast to those in [8] where robustness margins are introduced during abstraction for model inaccuracies. [5] uses a similarly motivated underlying idea to completely re-synthesize new robust strategies against a new GR(1) formula. Often uncertainties are not foreseen at the time of synthesis occur. In cases such as these, where unforeseen perturbations occur when the controller is implemented on the cyber-physical system, the results presented in this work allow for continued execution with guarantees in terms of formula satisfaction.

In summary, the main contributions of this work are the following: 1) to characterize the inherent tolerance of GR(1) strategies to unexpected perturbations; 2) to propose and prove approaches to refine GR(1) strategies to augment their tolerance to unexpected perturbations; 3) to quantify empirically the cost of augmenting the tolerance (robustness)

S. Dathathri and R. M. Murray are with the California Institute of Technology, Pasadena, CA, USA. S. C. Livingston is independent. Email addresses are sdathath@caltech.edu and {slivingston,murray}@cds.caltech.edu.

using the proposed approaches.

II. Preliminaries

For a finite set Σ, the set of all finite strings formed from concatenating elements of Σ is denoted by Σ^*, which is known as the Kleene closure [6]. The set of all countably infinite strings of Σ is Σ^ω. In this paper, a subscript notation is used, e.g., $\sigma_0\sigma_1\sigma_2\cdots\sigma_n \in \Sigma^*$, but observe that infinite strings can also be regarded as functions of the natural numbers $\mathbb{N}$ into Σ.

Let $\mathrm{AP}_{\mathrm{in}}$ be a set of input atomic propositions, and $\mathrm{AP}_{\mathrm{out}}$ be a set of output atomic propositions such that $\mathrm{AP}_{\mathrm{in}} \cap \mathrm{AP}_{\mathrm{out}} = \emptyset$. A state s is an assignment of `True` and `False` to the atomic propositions in $\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}}$. We use subset notation to indicate states and thus, for brevity, introduce $\Sigma = 2^{\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}}}$.

A *finite-memory strategy* is a pair (f, m_0) where $f : M \times 2^{\mathrm{AP}_{\mathrm{in}}} \to M \times 2^{\mathrm{AP}_{\mathrm{out}}}$ is a partial function and $m_0 \in M$, where $|M| < \infty$. Intuitively the set M represents the memory of the strategy. At each move, a new output is given depending on the input and the current memory value. As part of the move, a memory value is selected. Since we are only concerned with finite-memory strategies in this paper, we simply refer to them as *strategies*. The set of input-output sequences that may occur under f is defined as

$$\mathrm{Plays}(f) = \big\{\sigma \in \Sigma^\omega \mid \exists m \in M^\omega . \forall k \geq 0. \\ f(m_k, \sigma_k \cap \mathrm{AP}_{\mathrm{in}}) = (m_{k+1}, \sigma_k \cap \mathrm{AP}_{\mathrm{out}})\big\}, \quad (1)$$

where every $m \in M^\omega$ has the same first element, m_0. Elements of $\mathrm{Plays}(f)$ are referred to as *plays*. The set of prefixes that may be extended into a play is

$$\mathrm{Pref}(f) = \{\sigma \in \Sigma^* \mid \exists \alpha \in \Sigma^\omega . \sigma\alpha \in \mathrm{Plays}(f)\}\,. \quad (2)$$

Remark 1: For each $\sigma \in \mathrm{Plays}(f)$, there exists a unique $m \in M^\omega$ satisfying $f(m_k, \sigma_k \cap \mathrm{AP}_{\mathrm{in}}) = (m_{k+1}, \sigma_k \cap \mathrm{AP}_{\mathrm{out}})$ for $k \geq 0$.

It follows from the remark that a sequence of inputs determines precisely one output sequence.

We describe specifications for these strategies in linear temporal logic (LTL)[ref] in this paper. LTL formulae over propositions $(\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}})$ are evaluated over positions i in $\sigma = \sigma_0\sigma_1\ldots.. \in \Sigma^\omega$. In addition to the Boolean operators, the standard LTL operators $\Box$ (always), $\Diamond$(eventually) and $\bigcirc$(next) are used here for the specification.

A finite-memory strategy (f, m_0) is said to be

- *input-enabled* iff for every $\sigma^{\mathrm{in}} \in (2^{\mathrm{AP}_{\mathrm{in}}})^\omega$, there exists $\sigma \in \mathrm{Plays}(f)$ such that $\sigma_k^{\mathrm{in}} = \sigma_k \cap \mathrm{AP}_{\mathrm{in}}$ for $k \geq 0$.
- a *realization of* an LTL formula φ iff $\mathrm{Plays}(f) \subseteq L(\varphi)$ (also written as (f, m_0) *realizes* φ), i.e., for every $\sigma \in \mathrm{Plays}(f)$, $\sigma \models \varphi$.

A state $s \in \Sigma$ is said to be *reachable under* (f, m_0) iff there exists $\sigma \in \mathrm{Plays}(f)$ such that $\sigma_k = s$ for some $k \geq 0$.

A GR(1) formula is an LTL formula of the form

$$\Theta^{\mathrm{env}} \wedge \Box\rho^{\mathrm{env}} \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\psi_j^{\mathrm{env}}\right) \\ \implies \Theta^{\mathrm{sys}} \wedge \Box\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\psi_k^{\mathrm{sys}}\right), \quad (3)$$

where Θ^{env} is a state formula (i.e., without temporal operators) that is a function of $\mathrm{AP}_{\mathrm{in}}$, Θ^{sys} is a state formula that is a function of $\mathrm{AP}_{\mathrm{out}}$, and all ψ_j^{env}, ψ_k^{sys} subformulae are functions of $\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}}$ and also without temporal operators. The subformula ρ^{env} is a function of $\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}} \cup \bigcirc\mathrm{AP}_{\mathrm{in}}$, where

$$\bigcirc\mathrm{AP}_{\mathrm{in}} = \{\bigcirc x \mid x \in \mathrm{AP}_{\mathrm{in}}\}\,.$$

Except for $\bigcirc$ operators appearing as subformulae from $\bigcirc\mathrm{AP}_{\mathrm{in}}$, there are no other temporal operators in ρ^{env}. Finally, ρ^{sys} is defined similarly to ρ^{env} but as a function of $\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}} \cup \bigcirc\mathrm{AP}_{\mathrm{in}} \cup \bigcirc\mathrm{AP}_{\mathrm{out}}$.

To facilitate working with (3), and in particular the subformulae ρ^{env} and ρ^{sys}, we extend the semantics of the operator $\models$ for finite strings. Let $\sigma \in \Sigma^*$. Define

$$\sigma \models \rho \quad \iff \quad \sigma\alpha \models \rho \text{ for any } \alpha \in \Sigma^\omega, \quad (4)$$

where ρ is any Boolean formula that is a function of $\mathrm{AP}_{\mathrm{in}} \cup \mathrm{AP}_{\mathrm{out}} \cup \bigcirc\mathrm{AP}_{\mathrm{in}}$. Because at most one $\bigcirc$ operator binds to each atomic proposition, it follows that only σ_0, σ_1 determine whether the formula is satisfied.

Given a GR(1) formula φ as in (3), a state $s \in \Sigma$ is said to be *φ-reachable under* (f, m_0) iff there exists $\sigma \in \mathrm{Plays}(f)$ such that for some $k \geq 0$,

$$\sigma_k = s, \quad (5)$$
$$\sigma \models \Theta^{\mathrm{env}}, \quad (6)$$
$$\sigma_{j:(j+1)} \models \rho^{\mathrm{env}} \text{ for } j < k - 1. \quad (7)$$

A finite-memory strategy (f, m_0) is said to be a *strict realization* of (or to *strictly realize* a) GR(1) formula (3) if the following conditions are met

$$\sigma \models \Theta^{\mathrm{env}} \implies \sigma \models \Theta^{\mathrm{sys}} \quad (8)$$
$$\sigma \models (\Box^{-1}\rho^{\mathrm{env}} \implies \Box^{-1}\rho^{\mathrm{sys}}) \quad (9)$$

for any $\sigma \in \mathrm{Plays}(f)$. Intuitively, strict realizability ensures that blocking of an environment liveness condition when the other assumptions are met only occurs when the system is following tranition rules. Here, $\Box^{-1}$ is the Past LTL operator who semantics are as defined in [13].

III. Inherent Robustness of GR(1) strategies

Definition 2: A *perturbation* for a given finite-memory strategy is a deviant transition to a state s' in Σ from a state s, such that $f(m_j, s' \cap \mathrm{AP}_{\mathrm{in}}) \neq f(m_{j+1}, s' \cap \mathrm{AP}_{\mathrm{out}}) \vee ss' \not\models \rho^{\mathrm{env}}$.

A perturbation occurs when the system control action fails to drive the system to the state indicated by the strategy or the environment violates a safety assumption. In this section, we show that the GR(1) strategy with trivial refinement can satisfy a stricter formula - one that allows for finite perturbations when the transitions meet certain constraints. First we prove a lemma and which is then used to propose the refinement to allow for unexpected perturbations.

Let (f, m_0) be a finite-memory strategy that strictly realizes a GR(1) formula φ, let $\sigma \in \mathrm{Pref}(f)$ with $|\sigma| \geq 1$, and let $p^i \in \Sigma$. Let $\bar{I}$ be the set of φ-reachable states and $p^i \in \bar{I}$, $\gamma \in \Sigma^\omega$. Let $\tau^i\xi^i\alpha^i \in \mathrm{Plays}(f)$ where $\tau^i, \xi^i \in \Sigma^*$ for $i \in \{1, 2, ...n\}$.
Define 1_{jump} to be a formula such that for $\gamma_{j:j+1} \models 1_{jump}$ iff $f(m_{k+1}, \gamma_{j+1} \cap \mathrm{AP_{in}}) \neq f(m_{k+2}, \gamma_{j+1} \cap \mathrm{AP_{out}})$ where $k = j$ if $\mathbf{1}_0$ is True and $k = j - |\tau^i\xi^i_0| - \Sigma_{c=1}^{i-1}|\xi^c| - |\sigma|$ for $\mathbf{1}_i$ being True. Define φ^{jump} as below:

$$\varphi^{\mathrm{jump}} := \Theta^{\mathrm{env}} \wedge \Diamond\Box(\rho^{\mathrm{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\, \psi_j^{\mathrm{env}}\right) \implies$$

$$\Theta^{\mathrm{sys}} \wedge \Box\,\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\, \psi_k^{\mathrm{sys}}\right)$$

Lemma 3: If $\sigma_{-1}p^1 \models \rho^{\mathrm{sys}}$and $\xi^i_{-1}p^{i+1} \models \rho^{\mathrm{sys}}$ $\forall i \in 1, ..., n-1$. For any $\tau^i\xi^i\alpha^i \in \mathrm{Plays}(f)$ where $\tau^i, \xi^i \in \Sigma^*$ with $\xi^i_0 = p^i$ and $\tau^i\xi^i$ is a path through which p^i is φ-reachable, the following holds $\bar{\sigma} = \sigma\xi^1\xi^2...\xi^k\xi^{k+1}.....\xi^n\alpha^n \models \varphi^{jump}$.

Proof: The pre-ordered set of n-strategies chosen in Lemma 6 is chosen with replacement from the set of n strategies. This lemma arises as a direct consequence of Lemma 6 for the case where $m = 1$, that is the number of strategies synthesized is just 1. The same strategy is chosen n times and traces generated by the strategies are concatenated to generate a word satisfying φ^{jump} ■

Intuitively, the practical significance of Lemma 3 is that, if there is a disturbance that causes an unexpected transition to some state that is φ-reachable in other plays and if there are only finitely many such disturbances, then execution of the finite-state machine can continue after an appropriate change of its internal state and still result in a correct input-output sequence. This result also allows for actions even when ρ^{env} is violated. If ρ^{env} is violated during a particular transition between a state s and its successor s' i.e $ss'\neg \models \rho^{\mathrm{env}}$ and we end up at a φ-reachable state, this allows for a sequence of input-outputs that satisfies φ^{jump} if these disturbances occur in a finite manner. This suggests the refinement proposed subsequently. This result is useful because in practice once the symbolic computation during the synthesis of GR(1) strategies is done, only the φ-reachable paths are enumerated from the symbolic computation. In this instance, all the states stored are φ-reachable and only the ρ^{sys} condition must be checked before concatenating two paths and continuing further execution along the new path.

Consider a controller based on a GR(1) strategy. Consider the formula

$$\bar{\varphi} := \Theta^{\mathrm{env}} \wedge \Diamond\Box(\rho^{\mathrm{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\, \psi_j^{\mathrm{env}}\right) \wedge (\Diamond\Box\neg 1_{jump}) \implies \Theta^{\mathrm{sys}} \wedge \Box\,\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\, \psi_k^{\mathrm{sys}}\right). \quad (10)$$

Algorithm 1 with $n = 1$, generates an output sequence for a controller given an input sequence. The lemma above guarantees that this input-output sequence satisfies the formula $\bar{\varphi}$. It also considers for disturbances during application of the output action to a cyber-physical system.

The added $\Diamond\Box\neg 1_{jump}$ segment on the environment-assumption side in $\bar{\varphi}$ ensures that the perturbations do not occur infinitely often. And, for all instances of environment violation if a feasible φ-reachable state can be found, the system part of the formula is satisfied.
Also, we arrive at the following corollaries which help us augment the robustness of a given strategy strictly-realizing a GR(1) formula.

Corollary 4: Let (f, m_0) be a finite-memory strategy that realizes a GR(1) formula φ and φ^{jump} be the corresponding LTL formula as defined above. Let $\sigma \in \Sigma^\omega$, then

$$\sigma \models \varphi^{jump} \implies \sigma \models \varphi. \quad (11)$$

Proof: If $\sigma \models (\neg\Theta^{\mathrm{env}} \vee \Box\,\rho^{\mathrm{env}} \vee (\bigvee_{k=1}^{K} \Diamond\Box\neg\psi_j^{\mathrm{env}}))$ then $\sigma \models \varphi$. ■

Corollary 5: Let (f, m_0) be a finite-memory strategy that realizes a GR(1) formula φ, and let p be reachable under (f, m_0). If $\tau p\alpha \in \mathrm{Plays}(f)$ (at least one must exist), then $p\alpha$ satisfies

$$\Box\,\rho^{\mathrm{env}} \wedge \left(\bigwedge_{j=1}^{J} \Box\Diamond\, \psi_j^{\mathrm{env}}\right) \implies \Box\,\rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box\Diamond\, \psi_k^{\mathrm{sys}}\right). \quad (12)$$

IV. Augmenting Robustness

A. Approach 1: Concatenating multiple strategies with same safety/progress specifications

In this section, the intuition from Section III is used and a more general Lemma is presented and proved. The results in this section allow for the concatenation of multiple strategies synthesized with formulae differing in the initial condition and the approach for concatenation is described.

Given the specification φ_0 and a synthesized finite-memory strategy (f_0, m_0) that realizes φ_0, let $I(f_0, m_0)$ be the set of all states in the strategy.

Let η_0, η_1, $\eta_2 ... \eta_n$ be the additional states in Σ that the strategy must visit to provide additional robustness. Define $\eta_i^{in} = \eta_i \cap \mathrm{AP_{in}}$ and $\eta_i^{out} = \eta_i \cap \mathrm{AP_{out}}$. Define Θ_i^{env} as a Boolean formula which is True for a state s in Σ iff $s \cap \mathrm{AP_{in}} = \eta_i^{\mathrm{env}}$. Similarly, define Θ_i^{sys} as a Boolean formula which is True for a state s in Σ iff $s \cap \mathrm{AP_{out}} = \eta_i^{out}$.

Then, construct a set of finite-memory strategies (f_i, m_0^i) such that $\forall i \in \{0,1,2...,n\}$, (f_i, m_0^i) realizes φ_i, where φ_i is as defined below:

$$\varphi_i = \mathcal{T}_i^{in} \wedge \Box\, \rho^{\text{env}} \wedge (\bigwedge_{j=1}^{J} \Box \Diamond\, \psi_j^{\text{env}}) \implies$$
$$\mathcal{T}_i^{out} \wedge \Box\, \rho^{\text{sys}} \wedge (\bigwedge_{k=1}^{K} \Box \Diamond\, \psi_k^{\text{sys}}). \quad (13)$$

. Let (f_0, m_0^0) be a finite-memory strategy that strictly realizes a GR(1) formula φ_0, let $\sigma \in \text{Pref}(f_0)$ with $|\sigma| \geq 1$, and let $p^i \in \Sigma$.

Let there be a set of finite-memory strategies $\{(f_i, m_0^i)\}$ such that $\forall i \in \mathcal{I} = \{1, 2..., m\}$, (f_i, m_0^i) strictly-realizes φ_i, where φ_i is as defined above.
Consider a fixed ordering of the strategies, $\{f_{i_1}, f_{i_2}, f_{i_3}, ...f_{i_n}\}$ where $i_l \in \{1, 2...., m\}$ and $l \in \{1, 2...., n\}$. Let $\tau^l \xi^l \alpha^l \in \text{Plays}(f_{i_l})$ where $\tau^l, \xi^l \in \Sigma^*$. Define $\mathbf{1}_l$ and 1_{jump} as below: $1_{jump} \wedge \mathbf{1}_l \rightarrow \bigcirc(\mathbf{1}_{l+1} \wedge \neg \mathbf{1}_l)$. $\mathbf{1}_0$ is initialized to `True`.
Define 1_{jump} to be a boolean formula such that for $\gamma_{j:j+1} \models 1_{jump}$ iff $f_{i_l}(m_{k+1}, \gamma_{j+1} \cap \text{AP}_{\text{in}}) \neq f_{i_l}(m_{k+2}, \gamma_{j+1} \cap \text{AP}_{\text{out}})$ where $k = j$ if $\mathbf{1}_0$ is `True` and $k = j + |\tau^l| - \Sigma_{c=1}^{l-1} |\xi^c| - |\sigma|$ for $\mathbf{1}_l$ being `True`.

Lemma 6: If $\forall l \in \{1, ..., m\}$, p^l is φ_{i_l}-reachable under $(f_{i_l}, m_0^{i_l})$ through the path $\tau^l \xi^l$ with ${\xi^l}_0 = p^l$. And $\sigma_{-1} \xi_0^1 \models \rho^{\text{sys}}$, ${\xi^l}_{-1} \xi_0^{l+1} \models \rho^{\text{sys}}$ $\forall l \in \{1, ..., m-1\}$ then $\bar{\sigma} = \sigma \xi^1 \xi^2 .... \xi^n \alpha^n \models \varphi^{jump}$.
.
Proof: By definition, there exists $\beta^l \in \text{Plays}(f_{i_l})$ and k such that $p^l = \beta_k^l$, $\beta^l \models \Theta^{\text{env}_0}$, and

$$\beta_{j:(j+1)}^l \models \rho^{\text{env}} \text{ for } j < k-1. \quad (14)$$

Thus, we write $\tau^l \xi^l \alpha^l = \beta^l$ by taking $\tau_j^l = \beta_j^l$ for $0 \leq j < k$. And, $\xi_r^l = \beta_{k+r}^l$ where $0 \leq r < m_l$ for some m_l, and $\alpha_{j-k-m}^l = \beta_j^l$ for $j \geq k+m$. We want to show that $\sigma \xi^1 \xi^2 ... \xi^k \xi^{k+1} ..... \xi^n \alpha^n \models \varphi^{jump}$. Since φ^{jump} has the form of (3), this is equivalent to at least one of the following subformulae being satisfied: $\neg \Theta_0^{\text{env}}$, $\Diamond(\neg \rho^{\text{env}} \wedge \neg 1_{jump})$, $\Diamond \Box \neg \psi_j^{\text{env}}$ for some j, or

$$\Theta_0^{\text{sys}} \wedge \Box\, \rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond\, \psi_k^{\text{sys}} \right). \quad (15)$$

Since $\sigma \in \text{Pref}(f_0)$ by hypothesis, there exists $\gamma \in \Sigma^\omega$ such that $\sigma\gamma \in \text{Plays}(f_0)$. Also by hypothesis, (f_0, m_0) realizes φ_0, i.e., $\text{Plays}(f_0) \subseteq L(\varphi_0)$, hence $\sigma\gamma \models \varphi_0$. Since $|\sigma| \geq 1$ by hypothesis, $\sigma\gamma \models \neg \Theta_0^{\text{env}}$ if and only if $\bar{\sigma} \models \neg \Theta_0^{\text{env}}$. Thus, if $\sigma\gamma \models \neg \Theta_0^{\text{env}}$, then $\bar{\sigma} \models \varphi^{jump}$. Otherwise (i.e., if $\sigma\gamma \models \Theta_0^{\text{env}}$), consider the subformula $\Diamond(\neg \rho^{\text{env}} \wedge \neg 1_{jump})$. For all $k < |\sigma| - 1, \sigma_{k:k+1} \models \neg 1_{jump}$ by definition of a $\text{Play}(f_0)$ and that $\sigma \in \text{Pref}(f_0)$. Also, for $k \geq \bar{k} = |\sigma \xi^1 \xi^2 ... \xi^k \xi^{k+1} ..... \xi^n|, \alpha_{k-\bar{k}, k+1-\bar{k}}^n \models \neg 1_{jump}$ and $\xi_{-1}^n \alpha_0^n \models \neg 1_{jump}$ since $\tau^n \xi^n \alpha^n \in \text{Plays}(f_n)$. If $\sigma_{-1} \xi^1 \models \neg 1_{jump}$ and $\xi_{-1}^l \xi_0^{l+1} \models \neg 1_{jump}$ then $\bar{\sigma} \in \text{Plays}(f_0)$ and by definition $\bar{\sigma} \models \varphi^{jump}$ (since φ^{jump} reduces to φ_0 if 1_{jump} always evaluates to `False`). Consider the case when $\sigma_{-1} \xi_0^1 \models 1_{jump}$ and $\xi_{-1}^l \xi_0^{l+1} \models 1_{jump}$. For this case, we can conclude $\bar{\sigma} \models \Diamond(\neg \rho^{\text{env}} \wedge \neg 1_{jump}) \implies \big((\sigma\gamma \models \Diamond \neg \rho) \vee (\xi^1 \xi^2 ... \xi^k \xi^{k+1} ..... \xi^n \alpha^n \models \Diamond \neg \rho^{\text{env}})\big)$. If this is not the case, notice that we can trivially merge σ and ξ^1 or ξ^{l+1} and ξ^l repeatedly and satisfy the 1_{jump} condition.

If $\sigma\gamma \models \Diamond \neg \rho^{\text{env}}$, then there is a minimum k such that $\sigma_{k:}\gamma \models \neg \rho^{\text{env}}$ or $\gamma_{(k-|\sigma|):} \models \neg \rho^{\text{env}}$. If $k < |\sigma| - 1$, then $\bar{\sigma} \models \Diamond(\neg \rho^{\text{env}} \wedge \neg 1_{jump})$ (recall $\sigma_{k:k+1} \models \neg 1_{jump}$). Then $\bar{\sigma} \models \varphi^{jump}$.

From the hypothesis (φ_0-reachability and φ_l-reachability) and as a consequence strict-realization we have

$$\sigma_{-1} \xi_0^1 \models \rho^{\text{sys}}, \quad (16)$$

and

$$\xi_{-1}^l \xi_0^{l+1} \models \rho^{\text{sys}} \forall l \in 1, 2, 3..., n-1, \quad (17)$$

which we will refer to later while addressing the final case. The other case in which $\bar{\sigma} \models \Diamond(\neg \rho^{\text{env}} \wedge \neg 1_{jump})$ is if $\xi^l \models \Diamond \neg \rho^{\text{env}}$ for some l or $\xi^n \alpha^n \models \Diamond \neg \rho^{\text{env}}$ since ξ^l are themselves part of some play and $\sigma_{-1} \xi_0^1 \models 1_{jump}$ and $\xi_{-1}^l \xi_0^{l+1} \models 1_{jump}$. If any $\xi^l \models \neg \rho^{\text{env}}$ for $i \leq n-1$, then φ^{jump} is directly satisfied. If $\xi^n \alpha^n \models \Diamond \neg \rho^{\text{env}}$ then again φ^{jump} is satisfied.
Otherwise, suppose that $\bar{\sigma} \models \Diamond \Box \neg \psi_j^{\text{env}}$ for some j . From the semantics of LTL and the fact that ψ_j^{env} contains no temporal operators, this implies $\alpha^n \models \Diamond \Box \neg \psi_j^{\text{env}} \leftrightarrow \bar{\sigma} \models \Diamond \Box \neg \psi_j^{\text{env}}$. In this case, which again implies $\alpha^n \models \Diamond \Box \neg \psi_j^{\text{env}} \rightarrow \bar{\sigma} \models \varphi^{jump}$. We now consider the final case where $\bar{\sigma} \models \Theta_0^{\text{env}} \wedge \Diamond \Box(\rho^{\text{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \Box \Diamond\, \psi_j^{\text{env}}\right)$. By φ_n-reachability, $\tau_{d,d+1}^n \models \rho^{\text{env}}$ for all $d < |\tau^n| - 1$. And, $\xi^n \alpha^n \models \Box\, \rho^{\text{env}}$ as argued for this case (otherwise φ^{jump} would directly hold). Thus, $\tau^n \xi^n \alpha^n \models \Box\, \rho^{\text{env}}$. Recall that $\tau^n \xi^n \alpha^n \models \Theta_n^{\text{env}}$. Because (f_n, m^n) realizes φ_n by hypothesis and because $\tau^n \xi^n \alpha^n \in \text{Plays}(f)$ and $\tau^n \xi^n \alpha^n \models \Theta_n^{\text{env}}$, if neither $\tau^n \xi^n \alpha^n \models \Diamond \neg \rho^{\text{env}}$ nor $\Diamond \Box \neg \psi_j^{\text{env}}$ for any j, it must be that $\tau^n \xi^n \alpha^n$ satisfies φ_n. For this case, $\xi_{d,d+1}^l \models \rho^{\text{env}}$ $\forall d < |\xi^l| - 1$ (because otherwise φ^{jump} would directly hold).
By φ_l-reachability of ξ_0^l and strict-realizability, $\xi_{d,d+1}^l \models \rho^{\text{sys}}$ $\forall d < |\xi^i| - 1$. From (16) and (17), it follows that

$$\sigma_{-1} \xi^1 \xi^2 .... \xi^n \alpha \quad \models \quad \Box\, \rho^{\text{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond\, \psi_k^{\text{sys}} \right).$$

Recall the suffix γ such that $\sigma\gamma \in \text{Plays}(f_0)$. (f_0, m_0) strictly realizes φ_0, therefore if $\sigma\gamma \models \Theta_0^{\text{env}}$, it must be that $\sigma\gamma \models \Theta_0^{\text{sys}}$. $\sigma \models \Theta_0^{\text{sys}}$ since $\sigma \models \Theta_0^{\text{env}}$ (8). Furthermore, because in this case we are assuming there is no $0 \leq k < |\sigma| - 1$ such that $\sigma_{k:(k+1)} \models \neg \rho^{\text{env}}$ (otherwise we would have $\bar{\sigma} \models \varphi_0$), it follows from strict realizability (cf. (9)) that for $0 \leq k < |\sigma| - 1$, $\sigma_{k:(k+1)} \models \rho^{\text{sys}}$, and therefore $\bar{\sigma} \models \varphi^{jump}$. ∎

Consider GR(1) strategy based controllers, as discussed earlier, where only the φ-reachable states are retained and the environment moves are restricted to ones that do not violate ρ^{env}. Combine the finite-state controllers by adding transitions from all states to all other φ-reachable states that satisfy $ss' \models \rho^{\text{sys}}$. This combined set of controllers (strategies) will satisfy the formula $\bar{\varphi}$ with 1_{jump} being as defined in Lemma 6.

Algorithm 1 gives a formal description of the approach to combine strategies using the lemma proposed above. The notation used in the description is as defined in this section. This controller formed by the combined set of controllers (strategies) will satisfy the formula $\bar{\varphi}$ with 1_{jump} being as defined in Lemma 6.

Procedure 1 Implements controller based on Section IV

Input: finite-memory strategy (f_i, m_0^i) $\forall i \in \{1, 2, ..., n\}$, sequence of inputs $\sigma^{\text{env}} \in {\text{AP}_{\text{in}}}^{w}$, a system the control sequence σ^{sys} can be applied to and its state measured $s \in \Sigma$, set I - union of φ_i-reachable states for strategy f_i and M memory states corresponding to I and a mapping for every m in M to the strategy f_{i_m} it was taken from.
Output: Sequence of output actions $\sigma^{\text{sys}} \in {\text{AP}_{\text{out}}}^{\omega}$ satisfying $\bar{\varphi}$ when conditions in Lemma 6 are satisfied
memory=m_0
i=1
(memoryNew, σ_i^{sys}) = Strategy$f$: (memory, σ_0^{env})
safety=1
l=0
while (`True`) **do**
  if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 **then**
    (memoryNew, σ_i^{sys}) = Strategy f_l : (memory, σ_i^{env})
    Run: SafetyCheck
  else if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})({\sigma^{\text{env}}}_i) \not\models \rho^{\text{env}}$ OR safety=0 **then**
    if $\exists p$ and a corresponding m such that $p \in I$ and m in M and $i_m \in \{1, 2, ..., m\}$ and $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})p \models \rho^{sys}$ and $p \cap \text{AP}_{\text{in}} = {\sigma^{\text{env}}}_i$ **then**
      $({\sigma^{\text{env}}}_i, {\sigma^{\text{sys}}}_i) = p$, memoryNew=$m$, $l = i_m$
      Run: SafetyCheck
    else
      EXIT
    end if
  end if
  i+=1
  memory=memoryNew
end while

B. Approach 2: Augment Initial States

Let (f^0, m_0) be a finite-memory strategy that strictly realizes a GR(1) formula φ, with $|\sigma| \geq 1$, and let $p^i \in \Sigma$. Let σ belong to Pref(f^*) and $\sigma \models \Theta^{\text{env}} \wedge \Theta^{sys}$.

For a set of states $\eta \in \Sigma$ and $\eta \notin \text{I}(f, m_0)$, let χ^η be a Boolean formula indicating these states. Let (f^*, m_0^*) be a finite memory strategy that strictly realizes the formula φ^*

Procedure 2 SafetyCheck

apply σ^{sys_i}, measure s
if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})s \not\models \rho^{\text{sys}}$ **then**
  EXIT
end if
if $(\sigma_i^{\text{env}} {\sigma^{\text{sys}}}_i) = s$ **then**
  safety=1
else
  safety=0, σ_i^{env}=$s \cap \text{AP}_{\text{in}}$, σ_i^{sys}=$s \cap \text{AP}_{\text{out}}$
end if

that is defined as below:

$$\varphi^* := (\mathcal{I}_{in}^f \vee \chi_{in}^\eta) \wedge \Box \rho^{\text{env}} \wedge (\bigwedge_{j=1}^{J} \Box \Diamond \psi_j^{\text{env}})$$

$$\implies (\mathcal{I} \vee \chi^\eta) \wedge \Box \rho^{\text{sys}} \wedge (\bigwedge_{k=1}^{K} \Box \Diamond \psi_k^{\text{sys}}).$$

Corollary 7: If p_l is φ^*-reachable under (f^*, m^*) through the path $\tau^l \xi^l$ with ${\xi^l}_{-1} = p_l$ $\forall l \in 1, ..., m$, $\tau^l \xi^l \alpha^l \in$ Plays(f^*) , ssss$\sigma_{-1}\xi^1 \models \rho^{\text{sys}}$, ${\xi^l}_{-1}\xi_0^{l+1} \models \rho^{sys}$ $\forall l \in 1, ..., m-1$ then $\bar{\sigma} = \sigma \xi^{l_1} \xi^{l_2} .... \xi^m \alpha^m \models \varphi^{jump}$.
This corollary results as a special case of Lemma 6 with the number of strategies, $n = 1$ and picking a σ such that $\Theta^{\text{env}} \wedge \Theta^{\text{sys}}$ is satisfied. The utility in this approach is that if there a certain set of states that are recognized as states the system is likely to be perturbed to after executing the strategy, these states can be augmented to the initial-set of states visited by the strategy and using these as the initial states, synthesis can be done. This ensures that there is no loss of coverage in terms of the states visited by the initial strategy and there a single strategy that is robust to likely disturbances. Again, this strategy can be refined similarly as proposed earlier.

C. Approach 3: Patching

Let (f, m_0) be a finite-memory strategy that strictly realizes a GR(1) formula φ, let $\sigma \in \text{Pref}(f)$ with $|\sigma| \geq 1$, and let $p \in \Sigma$ and $p \notin \text{I}(f, m_0)$. Let η be an element in Σ and $\eta \notin \text{I}(f, m_0)$. Define $\mathcal{T}_{reach}$ as a Boolean formula that evaluates to `True` at a state s in Σ iff $s \in I(f)$. Let $(f_{reach}, \mathcal{M})$ be a finite-memory strategy that strictly realizes(defined similarly to the GR(1) specification) φ_{reach} where φ_{reach} is defined as

$$\left(\chi_{\eta^{in}} \wedge \Box \rho^{\text{env}} \wedge (\bigwedge_{k=1}^{K} \Box \Diamond \psi_k^{\text{env}})\right) \implies$$
$$\left(\chi_{\eta^{out}} \wedge \Box \rho^{\text{sys}} \wedge \Diamond \mathcal{T}_{reach}\right)$$

Define 1_{jump} to be a boolean formula such that for $\gamma \in \Sigma^*$, $j \geq 1$, $\gamma_{j:j+1} \models 1_{jump}$ if $f(m_j, \gamma_j \cap \text{AP}_{\text{in}}) \neq f(m_{j+1}, \gamma_j \cap \text{AP}_{\text{out}})$ and $\forall i < j, \gamma_{i:i+1} \models \neg 1_{jump}$. Otherwise, $\gamma_{j:j+1} \models \neg 1_{jump}$.

For $j = 0, \gamma_{j:j+1} \models 1_{jump}$ if $f(m_j, \gamma_j \cap \mathrm{AP}_{\mathrm{in}}) \neq f(m^k_{j+1}, \gamma_j \cap \mathrm{AP}_{\mathrm{out}})$. Otherwise, $\gamma_{j:j+1} \models \neg 1_{jump}$.

$$\varphi^{\mathrm{jump}} := \Theta^{\mathrm{env}} \wedge \Box(\rho^{\mathrm{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{j=1}^{J} \Box \Diamond \psi^{\mathrm{env}}_j \right) \implies$$

$$\Theta^{\mathrm{sys}} \wedge \Box \rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{sys}}_k \right)$$

Lemma 8: If p is φ_{reach}-reachable (defined similarly as for a GR(1) specification) under $(f_{reach}, \mathcal{M})$ and $\sigma_{-1}p \models (\rho^{\mathrm{sys}})$ then for $\sigma^{reach} \in \{\mathrm{Plays}(f_{reach})\}$ with the following properties holding:
(i) $\sigma^{reach} \models \Diamond \mathcal{T}_{reach}$
(ii) $k^* = min\{k : \sigma^{reach}_k = p\} < j^* = min\{j : \sigma^{reach}_j \models \mathcal{T}_{reach}\}$ and
(iii) σ_{j^*} is φ-reachable
(iv) $\sigma_{-1}p \models \rho^{\mathrm{sys}}$
then firstly, $(\sigma_{reach} \models \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{env}}_k\right)) \implies \sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \varphi^{jump}$ where $\gamma\sigma_{j^*}\alpha \in \mathrm{Plays}(f)$, $\gamma \in \Sigma*$. And secondly, $(\sigma_{reach} \models \left(\bigvee_{k=1}^{K} \Diamond \Box \neg\psi^{\mathrm{env}}_k\right)) \implies \sigma p \sigma^{reach}_{k^*+1:} \models \varphi^{jump}$.
Proof.
Let $\sigma^{reach} \in \mathrm{Plays}(f_{reach})$ such that it meets the four assumptions in the lemma and be the path through which p is φ^{reach}-reachable. For the case when , $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models (\bigvee_{k=1}^{K} \Diamond \Box \neg\psi^{\mathrm{env}}_j) \vee \Diamond(\neg\rho^{\mathrm{env}} \wedge \neg 1_{jump}) \vee (\neg\Theta^{\mathrm{env}})$ then $\sigma p \sigma^{reach}_{k^*+1:} \models \varphi$.

Consider the other scenario, $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \Theta^{\mathrm{env}} \wedge \Box(\rho^{\mathrm{env}} \vee 1_{jump}) \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{env}}_k\right)$. By φ^{reach}-reachability $\sigma^{reach}_{j:j+1} \models \rho^{\mathrm{env}}$ for all $j \leq (k^* - 1)$. If $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \Box(\rho^{\mathrm{env}} \vee 1_{jump})$, then $p \sigma^{reach}_{k^*+1:j^*}\alpha \models \Box \rho^{\mathrm{env}}$ (since $\sigma_{-1}p \models 1_{jump}$, see earlier proofs). Therefore, $\sigma^{reach}_{j:j+1} \models \rho^{\mathrm{env}}$ for all $j \leq (j^* - 1)$.

Case 1: $(\sigma_{reach} \models \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{env}}_k\right))$
We showed $\sigma^{reach}_{j:j+1} \models \rho^{\mathrm{sys}}$ for all $j \leq (j^* - 1)$. As earlier, by applying strict-realizability for f on σ, we get $\sigma_{j:j+1} \models \rho^{\mathrm{sys}}$ for all $j \leq |\sigma| - 2$. $\sigma_{-1}p \models \rho^{\mathrm{sys}}$.

By corollary 5, $\sigma_{j^*}\alpha \models \Box \rho^{\mathrm{env}} \wedge (\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{env}}_k) \implies \sigma_{j^*}\alpha \models \Box \rho^{\mathrm{sys}} \wedge (\bigwedge_{j=1}^{J} \Box \Diamond \psi^{\mathrm{sys}}_j)$. Therefore $\sigma^{reach}_{k^*+1:j^*}\alpha \models \Box \rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{sys}}_k\right)$, since it was shown that $\sigma^{reach}_{j:j+1} \models \rho^{\mathrm{env}}$ for all $j \leq (j^* - 1)$. Since $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \Theta^{\mathrm{env}} \wedge \Box \rho^{\mathrm{env}}$, by strict-realizability we get $\sigma_{l:l+1} \models \Box \rho^{\mathrm{sys}}$ and $\sigma \models \Theta^{\mathrm{sys}}$. Combined with our assumption $\sigma_{-1}p \models \rho^{\mathrm{sys}}$, we get $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \Theta^{\mathrm{sys}} \wedge \Box \rho^{\mathrm{sys}} \wedge \left(\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{sys}}_k\right)$. Therefore, $\sigma p \sigma^{reach}_{k^*+1:j^*}\alpha \models \varphi^{jump}$

Case 2: $\sigma_{reach} \models (\bigvee_{k=1}^{K} \Diamond \Box \neg\psi^{\mathrm{env}}_k)$
In this case, the intuition is that because we cannot invoke strict-realizability, we cannot give guarantees about satisfaction of ρ^{sys}. It is easy to see that $p\sigma^{reach}_{k^*+1:} \models (\bigvee_{k=1}^{K} \Diamond \Box \neg\psi^{\mathrm{env}}_k)$ given the assumption. Therefore, $\sigma p \sigma^{reach}_{k^*+1:} \models \varphi^{jump}$.

This result enables us to find a way to recover in the event of the environment violating a safety-specification on its part or the system failing to successfully transition to the state indicated by the original strategy (f, m) just building a patch to the original strategy. *Recover* here refers to the idea of satisfying the system part of the GR(1) specification. Though the proof presents the idea in the case of one such violation, the patching can be done recursively and the proof for that case would closely follow the one outlined above. During the recursive patching, the set of states to which the patch is built can be grown by augmenting the states reached in the initial strategy with the set of states visited in the previously patches build. This would likely grow coverage with each patch that is built. In addition, if disturbances(d) occurs to a φ_{reach^d}(defined similarly as above)-reachable state without breaking ρ^{sys} during the execution of the patch itself, it can also be treated similarly by building a new patch from the disturbed state to the total set of states visited. An algorithm for building patches and executing it on a system is formally stated in 3

D. Patching without progress

Instead of building a patch with the formula φ_{reach} specified above, one with the assumption on environmental progress relaxed can be used, $\varphi^{relaxed}_{reach} := \left(\chi_{\eta^{in}} \wedge \Box \rho^{\mathrm{env}} \wedge (\bigwedge_{k=1}^{K} \Box \Diamond \psi^{\mathrm{env}}_k)\right) \implies \left(\chi_{\eta^{out}} \wedge \Box \rho^{\mathrm{sys}} \wedge \Diamond \mathcal{T}_{reach}\right)$
$\varphi^{relaxed}_{reach}$ has lesser length than φ_{reach} and allows for faster computation. Since, $\varphi^{relaxed}_{reach} \implies \varphi_{reach}$, Lemma 7 holds with φ_{reach} replaced by $\varphi^{relaxed}_{reach}$.

V. Example Implementation and Analysis

Examples are implemented for the analysis of the techniques described in sections III,IV for the task of planar robot motion planning in the environments shown (See Figure 1). The robot is required to visit a set of locations infinitely often (progress-states) . A moving obstacle whose behavior dynamics mimic those of the robot with different progress-states and initial positions is added to the setup. The planned trajectories for the robot must be such that they do not collide with any of the walls (regions shaded black in Figure 1) or the non-deterministic moving obstacle.

A. Complexity for refinement

An empirical analysis of the computational costs involved in each of the approaches to augment robustness is presented here. The computations were peformed on a 2.40GHz Quadcore machine with 16 GB of RAM. The experiment described below is repeated 50 times and the average synthesis

Procedure 3 Algorithm for executing single patch from perturbed state

Input: GR(1) formula φ, finite-memory strategy (f, m_0), sequence of inputs $\sigma^{\text{env}} \in {\text{AP}_{\text{in}}}^{w}$, a system control sequence σ^{sys} can be applied and its state measured $s \in \Sigma$, set I of φ-reachable states for strategy f and M memory states, such that each $m \in M$ is the memory corresponding to some p in I along a φ reachable path.

Output: Sequence of output actions $\sigma^{\text{sys}} \in {\text{AP}_{\text{out}}}^{\omega}$ satisfying $\bar{\varphi}$ when conditions in Lemma 8 are satisfied

memory=m_0
i=1
l=0
(memoryNew, σ_i^{sys}) = Strategyf_l : (memory, σ_0^{env})
safety=1
while (`True`) **do**
  if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 **then**
    (memoryNew, σ_i^{sys}) = Strategy$f$: (memory, σ_i^{env})
    Run: SafetyCheck
  else if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})({\sigma^{\text{env}}}_{i}) \not\models \rho^{\text{env}}$ OR safety=0 **then**
    if Synthesize f^{reach} for φ^{reach} with $({\sigma^{\text{env}}}_{i}, {\sigma^{\text{sys}}}_{i})$ as intial state is successful **then**
      reached=0
      memoryNew=0
      while reached=0 **do**
        if $({\sigma^{\text{env}}}_{i-1}, {\sigma^{\text{sys}}}_{i-1})(\sigma_i^{\text{env}}) \models \rho^{\text{env}}$ and safety=1 **then**
          (memoryNew, σ_i^{sys}) = Strategyf_{reach} : (memoryNew, σ_i^{env})
          Run: SafetyCheck
          i+=1
          if safety=0 **then**
            EXIT
          end if
          if $({\sigma^{\text{env}}}_{i}, {\sigma^{\text{sys}}}_{i}) \in I$ **then**
            reached=0
            memoryNew= memory corresponding to $({\sigma^{\text{env}}}_{i}, {\sigma^{\text{sys}}}_{i})$ in M
          end if
        else
          EXIT
        end if
      end while
    else
      EXIT
    end if
  end if
  i+=1
  memory=memoryNew
end while

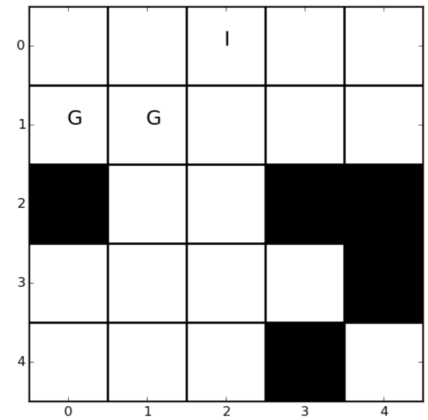

Fig. 1: Grid-World Setup

times are presented in Table I. Random 5x5 gridworlds are generated with a wall density of 0.2. The moving obstacle and robot have two different progress-locations which they visit infinitely often and two different initial positions. For each of the approaches 5 perturbation points are chosen as described:

- Multiple Strategy Approach: A single perturbation point is chosen that is not visited by the initial strategy and a strategy is synthesized. The states visited by the new strategy are stored. And, a new perturbation point is chosen not in any of the earlier strategies. This repeated 5 times.
- Patching: The points are chosen as the previous approach except only those states from the new strategy are stored that occurred before the trajectories hit the old set of states.
- Patching without progress: Similarly done as patching, if synthesis from a perturbed point is not feasible, another point is chosen. This is done so till a point is found from which a feasible patch exists.
- Augmented Initial States Approach: Five arbitrary points not visited by the original strategy are chosen and a new strategy is synthesized.

The coverage i.e the number of unique states - robot, moving obstacle position combinations - visited by each strategy is also presented. It loosely characterizes the robustness for the concatenated-strategies as this count represents the φ-reachable states. Approach 3 is implemented recursively, with the visited states augmented in each patch. With recursive patching, the time for synthesis tends to decrease progressively with each patch for a given gridworld because the number of unique visited states tends to go up. Also, the numbers indicate that the synthesis for patching without the progress condition is faster than that with progress, as expected because the synthesis formula has smaller length.

Approach for refinement	Average time for synthesis(s)	Coverage (unique states)
New strategy from perturbed state	0.22	145.14
Augment Initial States	0.21	144.92
Patching	2.98	173.63
Patching without progress	1.90	130.36

TABLE I: Runtimes and unique states visited

VI. Conclusion and Future Directions

We demonstrated the inherent tolerance of strategies synthesized to satisfy GR(1) specifications and described approaches to augment the tolerance of a strategy to perturbations by refinement or concatenation with other strategies in a provably correct manner. It was shown that these refined strategies satisfy a stricter formula than the one used for synthesis. This tolerance is useful when the model is not exact for either the system behavior or the environment behavior.

In the future, we plan to extend the framework built here to the case of infinite jumps. We also intend to develop a metric that would quantify the robustness added to a strategy through a given concatenation and prescribe approaches for refinement of strategies to make them more robust with optimal synthesis time/memory costs. Also, we plan to implement the approaches in Sections IV-A and IV-B using enumeration from a stored BDD computed during the original synthesis as that would remove the need for re-synthesis to obtain the new strategies. This is not directly facilitated by the solver 'gr1c' [9] used for the work presented here.

Acknowledgments

This work was partially supported by United Technologies Corporation and IBM, through the industrial cyber-physical systems (iCyPhy) consortium.

References

[1] R. Bloem, K. Chatterjee, K. Greimel, T. A. Henzinger, and B. Jobstmann. *Computer Aided Verification: 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings*, chapter Robustness in the Presence of Liveness, pages 410–424. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[2] R. Bloem, K. Greimel, T. A. Henzinger, and B. Jobstmann. Synthesizing robust systems. In *Formal Methods in Computer-Aided Design, 2009. FMCAD 2009*, pages 85–92, Nov 2009.

[3] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Sa'ar. Synthesis of reactive(1) designs. *Journal of Computer and System Sciences*, 78:911–938, May 2012.

[4] R. Ehlers. *NASA Formal Methods: Third International Symposium, NFM 2011, Pasadena, CA, USA, April 18-20, 2011. Proceedings*, chapter Generalized Rabin(1) Synthesis with Applications to Robust System Synthesis, pages 101–115. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[5] R. Ehlers and U. Topcu. Resilience to intermittent assumption violations in reactive synthesis. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pages 203–212, New York, NY, USA, 2014. ACM.

[6] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[7] Y. Kesten, N. Piterman, and A. Pnueli. Bridging the gap between fair simulation and trace inclusion. *Information and Computation*, 200:35–61, 2005.

[8] J. Liu and N. Ozay. Abstraction, discretization, and robustness in temporal logic control of dynamical systems. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pages 293–302, New York, NY, USA, 2014. ACM.

[9] S. C. Livingston. gr1c: a collection of tools for GR(1) synthesis and related activities. `http://scottman.net/2012/gr1c`. [Online; accessed 15-March 2016].

[10] R. Majumdar, E. Render, and P. Tabuada. Robust discrete synthesis against unspecified disturbances. In *Hybrid Systems: Computation and Control (HSCC)*, April 2011.

[11] Z. Manna and A. Pnueli. A hierarchy of temporal properties. In *(PODC '90) Proceedings of the ninth annual ACM Symposium on Principles of Distributed Computing*, pages 377–408, 1990.

[12] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag New York, Inc., New York, NY, USA, 1992.

[13] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, SFCS '77, pages 46–57, Washington, DC, USA, 1977. IEEE Computer Society.

[14] M. Rungger and P. Tabuada. A symbolic approach to the design of robust cyber-physical systems. In *Proceedings of the 52nd IEEE Conference on Decision and Control, CDC 2013, December 10-13, 2013, Firenze, Italy*, pages 3932–3937, 2013.

[15] M. Rungger and P. Tabuada. Abstracting and refining robustness for cyber-physical systems. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pages 223–232, New York, NY, USA, 2014. ACM.

[16] D. C. Tarraf, A. Megretski, and M. A. Dahleh. A framework for robust stability of systems over finite alphabets. *IEEE Transactions on Automatic Control*, 53(5):1133–1146, June 2008.

[17] G. Zames. Input-output feedback stability and robustness, 1959-85. *IEEE Control Systems*, 16(3):61–66, Jun 1996.