We thank the reviewers for their thorough reviews, valuable feedback and great questions. Multiple reviewers suggested as an improvement to release the code for reproducibility. We intend to release the code for RTAC, our version of SAC, as well as all the code necessary to reproduce the MuJoCo and car simulation experiments.

We realize that we might not have convincingly communicated the usefulness of the real-time framework beyond modelling action selection time. We try to address this in our answer to Reviewer 1. However, before that, we want to clarify the central assumption of the paper, since the relevant paragraph from line 69 ff. seemed to have been quite confusing. There are two different time spans:

timestep size	t_s	(the time between two observations)
action selection time	t_π	(e.g. time required for a forward pass through the policy network)

RTRL deals with the special case in which $t_s = t_\pi$. In that case an action a_t does not affect the next state s_{t+1}, which opens up a number of new algorithmic possibilities. We think $t_s = t_\pi$ is the right assumption because it leads to *back-to-back action selection*. That is, immediately upon finishing to compute an action the next observation is sampled. This should always be the goal, no matter how little time is required to compute an action. It allows the agent to update its actions the quickest, e.g. if we could compute an action in 1ms we should do so 1000 times per second.

Review 1 We agree, that an experiment with different action selection times for SAC would be interesting. It should be noted though, that RTAC (with $t_\pi = t_s$) even outperforms the most idealized ($t_\pi = 0$) version of SAC (see Fig. 9 in the Appendix). One might argue that this comparison between RTAC and SAC is irrelevant because they are different algorithms. However, the only differences between SAC and RTAC are those made possible by using the real-time framework, which is actually exactly what we think this paper is about. We believe this line of work is valuable because by assuming $t_\pi = t_s$ the framework becomes simpler rather than more complex (compare MRP and RTMRP) and it gives us the opportunity to create better algorithms.

We used the transformation notation to be able to establish equivalences such as $\mathit{RTMRP}(E, \boldsymbol{\pi}) = \mathit{MRP}(\mathit{RTMDP}(E), \boldsymbol{\pi})$. However, we agree, that the notation is inconsistent. We refer to both the function and to its result as MRP. We plan to rename the standard, turn-based MRP-function to TBMRP. The transformation TB will be called TBMDP indicating that, like RTMDP, it is a function transforming one MDP into another. The two central equivalences, showing that real-time interactions can be expressed with turn-based interactions and vice versa, would then look like

$$\mathit{RTMRP}(E, \boldsymbol{\pi}) = \mathit{TBMRP}(\mathit{RTMDP}(E), \boldsymbol{\pi}) \quad \text{and} \quad \mathit{TBMRP}(E, \pi) \propto \mathit{RTMRP}(\mathit{TBMDP}(E), \boldsymbol{\pi}).$$

For the right side we will add proper definitions and proof in the appendix. So far, we neglected the TBMDP a bit since the rest of the paper does not depend on it. As suggested, we will rename u to a.

Review 2 If we understand it correctly, the question about jitter refers to the problem of aliasing in case we are not able to sample quickly enough the quantities that we want to observe (Nyquist-Shannon sampling theorem). Both with turn-based and with real-time interaction, we might run into the problem of not being able to further reduce the timestep size beyond the action selection time of our agent. In that case, information could "hide" in higher, unobserved frequencies making the observation process non-markov and thus not representable by a MDP. Especially if our environment is non-markov for other reasons too, one solution might be to transform the observation process into a markov process by concatenating all past observations. Another, more complex solution, would be to reduce the policy evaluation time by breaking the policy up into a pipeline of stages (as mentioned in line 79). All stages could be evaluated in parallel, reducing computation time and allowing us to lower the timestep size which would solve our aliasing problem. More precisely, we would construct a "pipelining MDP" where an action would contain the outputs of all the pipeline stages and a state would contain the inputs to all the stages. The transition function of the pipelining MDP would simply forward the intermediate computations from one stage to the next (in addition to representing the actual, underlying state transitions).

We only directly compare RTAC with SAC. With other algorithms it would not have been a like-for-like comparison. However, we will consider adding TD3 (the only other competitive algorithm) to our comparison and codebase.

Review 3 We motivate our paper as being one of the first reformulations of RL to take the *time to select an action* into consideration, not the "time of the action". We introduce and mathematically define the RTMRP framework, an interaction framework that assumes that the agent takes exactly one timestep to select an action. We furthermore explain why the assumed time of a single timestep is a reasonable choice and why we believe it does not limit generality. The RTRL framework stands in contrast to the conventional RL interaction framework, in which the agent selects the action for the current timestep, and no time is allowed for selecting the action.

We hope that this response helps Reviewer 3 to see how the content of our paper matches its motivation. Since this was their only negative comment, we would hope that, with this important distinction being underscored, they might reassess their "reject" vote.