We thank the reviewers for their insightful comments. We summarize the questions from each reviewer below and address them separately. Due to the space limit, we try our best to answer all the major questions from each reviewer.

(Reviewer 1) Q1: Justify why SMAC is generalizable. TMC may fail for chaotic MAS.

A: SMAC is a challenging benchmark that has been used by recent works to evaluate their MAS communication schemes [33,37,8,20]. Unlike other environments, SMAC offers partial observability, challenging dynamics, complex observation and high-dimensional action, which closely simulates the practical scenarios. TMC can perform well under a chaotic MAS by eliminating noisy information in the observation. Please refer to Q1 of reviewer 2 for more details.

(Reviewer 2) Q1: TMC may not applicable to the environment where the observation is not smooth.

A: TMC works even if the observation is chaotic. TMC can eliminate the noisy information in the observation, making message exchange occur only if additional and useful information is presented in the current observation. To show this, we attach the observation $\mathbf{o}_t$ with a vector of Gaussian noise for 6h_vs_8z scenario, and train TMC with this new observation $\mathbf{o}'_t$. Figure (a) shows the pdf of $||\mathbf{o}_t - \mathbf{o}_{t-1}||$,$||\mathbf{o}'_t - \mathbf{o}'_{t-1}||$,$||\mathbf{m}_t - \mathbf{m}_{t-1}||$ and $||\mathbf{m}'_t - \mathbf{m}'_{t-1}||$ across time t, where $\mathbf{m}_t$ and $\mathbf{m}'_t$ are the messages generated with $\mathbf{o}_t$ and $\mathbf{o}'_t$, respectively. We see that the observation becomes very bumpy after introducing the additional noise, but the generated messages are still smooth (*i.e.*$||\mathbf{m}'_t - \mathbf{m}'_{t-1}||$ is small in general). Figure (b) depicts the corresponding impact on TMC training process, although the noise slows down the training initially, TMC can still converge to the same result finally. We will revise the paper to clarify this point.

(Reviewer 2) Q2: How the regularizer in Eq. 2 gives robustness against message variation and affects Q-values?

A: Let's use an example to clarify. Assume the output of local action generator (local Q-values) at receiver agent has two elements, denoted as $\mathbf{Q} = [q_1, q_2]$. Assume current message $\mathbf{n} = [n_1, n_2]$ at sender and stored message $\mathbf{m} = [m_1, m_2]$ at receiver are close in l_2 distance, so $\mathbf{n}$ will not be transmitted by sender. To infer action, receiver performs elementwise addition between $\mathbf{Q}$ and $\mathbf{m}$, giving global Q-values $[q_1 + m_1, q_2 + m_2]$. Yet, the more accurate global Q-values should be $[q_1 + n_1, q_2 + n_2]$. Let $e^1 = q_1 + m_1$ and $e^2 = q_2 + m_2$ denote the largest and second largest Q-values. Eq2 encourages $e^1 - e^2$ to be large (*i.e.*much greater than 0). Therefore it is highly likely that $e^1 - e^2 > (n_2 - m_2) - (n_1 - m_1)$ (since $\mathbf{n}$ and $\mathbf{m}$ are close, $(n_2 - m_2) - (n_1 - m_1)$ is close to 0). With some derivations, we get $q_1 + n_1 > q_2 + n_2$. That is, the regularizer in eq2 allows a better chance of inferring the correct receiver action with the stored message $\mathbf{m}$.

(Reviewer 2) Q3: Why perform quantization? What is c_1^t? How global observations and histories are obtained?

A: We quantize the messages to further reduce communication overhead. c_1^t is the intermediate result of action generator, which also is the input of message encoder. Global observation and histories are obtained only through the messages.

(Reviewer 3) Q1: Why not let receiver to initiate the communication? Why not use TCP?

A: Receiver-initiated scheme causes a higher communication overhead and delay, as the receiver needs to first send the request to senders before getting the messages. This also causes a higher chance for loss, as the communication is failed if either request or message is lost. In contrast, TMC requires to send the message only, which mitigates above issues. TCP ensures reliable communication by retransmitting the lost packets until delivered successfully, which is not suitable for the real-time MARL execution, where agents need to use the messages instantly within the same timestep. Packet retransmission also causes a large communication overhead and delay, which seriously impairs MARL performance.

(Reviewer 3) Q2: Why halting training processing? Does TMC+QMIX still outperform AC after convergence?

We actually pause the training and run 20 test episodes to evaluate the current trained agent network. We then resume the training process until convergence. As shown in Figure 4 in the paper, TMC+QMIX also wins AC after convergence.

(Reviewer 3) Q3: Ablation studies should be performed on more hyperparameters (*e.g.*λ_s, λ_r,σ, w_s, β).

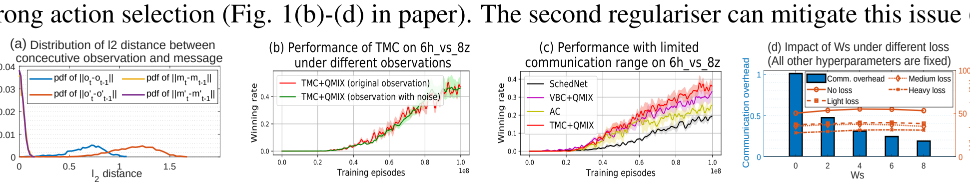
A: The ablation studies for λ_s, λ_r and σ are given in the appendix. Figure (d) and Table (e) below show the impact of w_s and β on TMC performance for 6h_vs_8z. For w_s, although $w_s = 8$ achieves a smaller communication overhead, $w_s = 6$ gives better winning rates under different loss patterns. For β, $\beta_1 = 1.5, \beta_2 = 1, \beta_3 = 0.5$ give a slightly better winning rate, and the communication overhead does not vary too much for the different selection of $\beta_1, \beta_2, \beta_3$.

(Reviewer 3) Q4: Author should consider the scenario where agents have a limited communication range.

A: We perform a new evaluation by restricting the agents to communicate only within a *communication range*. Figure (c) below shows the winning rates for 6h_vs_8z by setting communication range to 9, and keeping the agent observation range to 6. We can see that TMC+QMIX still beats the other methods. This indicates that TMC can always remove the noisy information in the messages to accelerate the training, even if the communication range is limited.

(Reviewer 4) Q1: Why do you choose SMAC for evaluation? What does the second regulariser do?

A: SMAC has been used by recent works for testing their communication schemes in MARL [37,33,20,8]. In SMAC, due to partial observability of each agent, communication plays a key role in learning the advanced techniques (*e.g.*kiting, besiege), making it a good benchmark for TMC. We will evaluate TMC on more environments (*e.g.*ParlAI) in the future. During execution, the small difference between the stored message at receiver and true message at sender may cause the wrong action selection (Fig. 1(b)-(d) in paper). The second regulariser can mitigate this issue (See Q2 of reviewer 2).



(e) Impact of β (assume no message loss, other hyperparameters are fixed)

$[\beta_1, \beta_2, \beta_3]$	Win rate	Comm. overhead
$[1.5, 1.0, 0.5]$	55.4%	$1\times$
$[1.2, 1.0, 0.7]$	55.0%	$0.98\times$
$[1.0, 1.0, 1.0]$	54.5%	$0.97\times$
$[0.5, 1.0, 1.5]$	52.7%	$1.05\times$