IRC 25 report

This report by Rakshith Raj addresses issues and mistakes found in the autonomous domain. While it primarily focuses on autonomous domain, it will include some relevant issues from other domains. The report presents a structured list of firstly the anecdotes and then their problems and their corresponding solutions.

1. The Training sessions

The training sessions were brief, near and clean. I admire the efforts of my seniors and respect them a lot. The sessions that were taken are:-

→ **Git and GitHub**: This session was all about the commands to run and showed only the highest level of git and GitHub.

**Problem**: The problem is that everyone knows the commands to run and do the task but no one knew how to use git and GitHub efficiently. After this session we had brief attempts to use git and GitHub but eventually we would just end up using WhatsApp. This is not at all acceptable.

**Solution**: We have to take a git and GitHub session and teach the juniors on how to use it rather than just give out commands. We should make sure everyone uses GitHub right from the go. When we make a private repo, i understand that it takes a little effort to setup ssh keys and all to clone and push it but isn’t that the effort we are have to do to save our codebase.

→ **React and Node** : I do not know why this session was taken at all. We did neither touched react not did we touch node in this whole year. We had 2 whole sessions on this.

**Problem**: The sessions which are planned and taken or not relevant to the work we were going to do. I get it we used the website last year for communication but still.

**Solution**: There should be a meticulous plan on what we were going to use in IRC and based on that what to teach. Technologies which are sure shot going to be used should be taught in the sessions. If we are dicey on whether to take the session or not because we might not use it. Don’t take the session. If in the future we might need it, we can spare 2 days or have night labs for teaching or learning the tech on our own.

**Overall solution for segment**: Plan sessions based only on technologies we will definitely use. Skip sessions for technologies we’re uncertain about using. The issue isn’t with learning itself—it’s about using our time wisely. We should focus our time on learning the tools and technologies that will directly benefit our work.

1. Starting 2 weeks of night lab

The starting 2 weeks of our night lab were uneventful. We didn’t do much work related to our final product. We were asked to look through stuff that we didn’t use at all. That is not the problem at all. I don’t mind looking through and learning.

**Problem**: the problem arises when we are actually spending less time on the actual main stack.

**Solution**: A plan on what the juniors should do should be there when sessions are over or before the juniors arrive for night labs. At least the task list should be planned. The assigning of tasks can be done based on the juniors performance in the training sessions.

1. Instructing juniors to go for ARC when IRC isn’t over

During our holiday break, the entire autonomous team gathered on Google Meet. We were instructed to prepare advanced topics and refine existing ones for upcoming juniors and ARC. We were told not to worry about the IRC code since the senior core would handle it.

**Problem**: I don’t need to specify what the problem here is. Why were we asked to work on something that is not going to happen for another 5 to 6 months. Learning advanced topics would not take more than 1 1.5 months which we easily have after IRC. The bigger problem is the senior core have barely worked on the code till then. After we came from the holidays the stack stopped working and pradeep and I ended spending 2 days fixing it.

**Solution**: We should hold weekly meetings to review progress and share what everyone has learned. This ensures the whole team stays informed about both learning progress and codebase changes. We should focus on one competition at a time.

1. Making a lot of changes in the last month of IRC

After the 10 days Diwali holidays the code base went through a lot of changes. Like a lot. Even during IRC we were doing changes.

**Problem**: The problem in changing the codebase in the last one month of IRC is testing. Without testing we cannot finalise the code. We will not have confidence. We can only hope the code works.

**Solution**: Fix the code at least 1 month before IRC and in the month only testing should happen. Small changes like variable changes, range changes and stuff like that. This will enable us to rest the code intensively. Especially during IRC there should not be any changes at all. Till IRC do testing make changes small and medium size but after IRC fix the code completely.

1. Suggestions given to seniors

I remember i gave a suggestion of using a certain technique to enhance the codebase. Many of our junior core also gave suggestion to implement. In fact I remember giving it multiple times.

**Problem**: We come to the assumption that our seniors aren’t considering our suggestions and this creates a bad rapport with the seniors.

**Solution**: Tell us why the suggestion won’t suffice.

1. Code changes and integration

I was given the task of making the obstacle avoidance code. I spent 3 4 days on it brain stormed a lot and sent a half baked. It still worked for one obstacle. I sent it as I didn’t need completely bake it for IRC.

**Problem**: the code I sent and the code that was integrated was completely different. If had no relation to what I sent. I sat for 3 days and wrote the code which was completely changed. Later during IRC I was asked to fix the code as it was not working.

**Solution**: any code change should be done in the presence of the code author or the person changing the code should ask the code author to explain his/her code and then do the change.

1. Planning

I have noticed from the very beginning(excluding the first few weeks) that planning was always done in discrete and then with a ready plan we were just presented on what to do.

**Problem**: The juniors wouldn’t get a chance to be involved in the planning.

**Solution**: involve juniors in planning, they could have more ideas or better ones. At the end we can end up having way better plan with more detailed steps.

Suggestions for other domainso

- Make everything in the rover modular. Components should be designed with a “plug-and-play” approach for easy integration and use. Like this IRC we won’t have to worry about detaching the robotic arm if we have less time if made modular.

- Consider thermal management in mechanical design proper heat dissipation pathways is crucial for the realsense, laptop