* As technology improving, there are more and more application of autonomous mobile robots in different fields, such as in factories, logistics and hospitals. One of the biggest advantages of mobile robots is the ability to transport items and move automatically without direct human control. This not only saves on labor costs but also reduces so much tedious work that people have to do. Because of this, improving the autonomy of AMRs can greatly expand their use, allowing more people to benefit from their capabilities. So, to achieve this, I focus on Motion planning for AMRs, which plays a crucial role in enabling their autonomy.
* What is Motion planning in the field of robotics? Typically, it finds a continuous sequence of valid configurations that safely and efficiently guide a robot from start to goal. There are two critical parts: global planning and local planning. My research mainly focuses on local planning, but I need to introduce both first.

Global planning is like navigation in Google Maps; it creates a clear route from the starting point to the goal, telling robots how to avoid static obstacles. On the other hand, local planning must handle all emergencies in real time. It generates a motion command in every short time period that helps robots evade dynamic objects, just like a driver constantly making decisions on the go. Therefore, a better local planning system should help AMRs navigate highly dynamic and complex environments, consequently improving their autonomy.

* There are several methods for solving local planning problems. Artificial Potential Fields (APF) model obstacles as repulsive forces and goals as attractive forces, allowing mobile robots to avoid obstacles in real time. However, they often struggle with local minima issues.

Dynamic Window Approach (DWA) samples velocity commands within dynamic constraints and scores them based on clearance, velocity, and heading. While effective, it struggles with continuously moving obstacles. In fact, one of our former lab members focused on addressing this issue.

Velocity Obstacles (VO) predict collision regions in velocity space and select safe velocities. This method is highly effective for multi-agent systems but struggles to avoid moving obstacles without prior knowledge of their intentions.

Model Predictive Control (MPC) optimizes trajectories over a short horizon while respecting dynamic constraints. It effectively handles robot dynamics and obstacle avoidance but can become computationally expensive as the planning horizon increases.

My research aims to combine the Velocity Obstacle method with the concept of MPC to enhance the avoidance capabilities of autonomous mobile robots (AMRs).

* Specifically, I seek to determine the optimal sequence of motion commands within a short time horizon to handle obstacles with unknown intentions. This process is structured into two key components: **Prediction and Optimization**, each playing a crucial role in improving AMRs’ navigation performance in dynamic environments.
* In the prediction step, we aim to estimate the movements of surrounding obstacles—specifically, their future positions and velocities. To achieve this, I assume that the positions and velocities of all obstacles can be measured and model each observed obstacle as a constant acceleration agent, meaning they maintain the same acceleration over a short time period.

To refine these predictions, a Kalman Filter is applied to estimate the confidence level of the predicted trajectories. Typically, the longer the prediction horizon, the greater the uncertainty, leading to higher variance in the estimates. As a result, the farther into the future the prediction extends, the larger the predicted uncertainty, which in turn expands the estimated danger region of the obstacle.

* Before moving into the optimization step, I need to introduce the Velocity Obstacle (VO) method. If an agent A encounters an object B, we can identify a danger region for A's movement. How? Intuitively, we draw two tangent lines from A's center to the inflation circle of object B in the configuration space. This forms a sector region indicating that moving in this direction would be dangerous for agent A.

Next, we change our perspective. By mapping this sector into velocity space, the region now represents the set of relative velocities (v<sub>AB</sub>) that would lead to a collision. This is known as the Collision Cone.

When object B has its own velocity, we need to determine which velocities for A are safe or dangerous. To do this, we compute the Minkowski sum of the Collision Cone and B's velocity vector. The resulting region indicates all the dangerous velocities for agent A and is referred to as the Velocity Obstacle.

* After defining the Velocity Obstacle (VO), an AMR has to find the optimal motion that avoids obstacles while still progressing toward its goal. Essentially, this becomes an optimization problem.

Conventional VO methods determine the robot's next motion at each time step and continuously adjust it in the following optimization cycle. The design variable in this approach is the robot’s next motion. Algorithm’s goal is to find the feasible velocity closest to the ideal velocity. To achieve this, a VO constraint function is defined. If a motion falls within the VO region, the function's value will be greater than zero, indicating a potential collision. The optimization problem is therefore subject to the condition that the function value must be less than zero, ensuring collision-free motion. Thus, the optimal motion for the robot's next step will lie within the dynamic feasible region but outside the VO.

cost(\mathbf{v}\_{next})|\_{\mathbf{v}\_{ideal}}=||\mathbf{v}\_{next}-\mathbf{v}\_{ideal}||

**{\color{**Blue**}** define**}** **\begin{**cases**}** **\mathfrak{**VO**}\_**B**(\mathbf{**v**}\_{**next**}\in** VO**\_{**AB**})**>**0\\** **\mathfrak{**VO**}\_**B**(\mathbf{**v**}\_{**next**}\notin** VO**\_{**AB**{\color{**Blue**}** **{\color{**Blue**}** **}}})\leq0** **\end{**cases**}**

**{\color{**red**}**subject\ to\ **}\begin{**cases**}** **\mathfrak{**VO**}\_**B**(\mathbf{**v**}\_{**next**})\leq0\\** \**|\mathbf{**v**}\_{**next**}**\**|\leq\max**\ v **\end{***cases***}**

min\ Obj**(**cost**\_1+**cost**\_2)**

**\begin{**aligned**}** **&**cost**\_1(\mathbf{**v**}\_1**,**\mathbf{**v**}\_2**,**\dots**,**\mathbf{**v**}\_**i**)|\_{\mathbf{**p**}\_**w**}\\** **&**=\**|\mathbf{**p**}\_1(\mathbf{**v**}\_1)-\mathbf{**p**}\_**w\**|+**\**|\mathbf{**p**}\_2(\mathbf{**v**}\_1**,**\mathbf{**v**}\_2)-\mathbf{**p**}\_**w\**|+\dots+**\**|\mathbf{**p**}\_**i**(\mathbf{**v**}\_1**,**\mathbf{**v**}\_2**,**\dots**,**\mathbf{**v**}\_**i**)-\mathbf{**p**}\_**w\**|\\** **&**=**\sum\_{**k=**1}^**i\**|\mathbf{**p**}\_**i**-\mathbf{**p**}\_**w\**|** **\end{***aligned***}**

**\begin{**aligned**}** **&**cost**\_2(\mathbf{**v**}\_1**,**\mathbf{**v**}\_2**,**\dots**,**\mathbf{**v**}\_**i**)|\_{\mathbf{**v**}\_0}\\** **&**=\**|\mathbf{**v**}\_1-\mathbf{**v**}\_0**\**|+**\**|\mathbf{**v**}\_2-\mathbf{**v**}\_1**\**|+\dots+**\**|\mathbf{**v**}\_**i**-\mathbf{**v**}\_{**i**-1}**\**|\\** **&**=**\sum\_{**k=**1}^**i\**|\mathbf{**v**}\_**i**-\mathbf{**v**}\_{**i**-1}**\**|** **\end{***aligned***}**

**{\color{**Red**}** subject\ to\ **}\begin{**cases**}** **\mathfrak{**VO**}\_{**B,**1}**,**\mathfrak{**VO**}\_{**B,**2}**,**\mathfrak{**VO**}\_{**B,**3}**,**\dots**,**\mathfrak{**VO**}\_{**B,i**}\leq0\\** \**|\mathbf{**v**}\_**i\**|\leq\max**\ v **\end{***cases***}**

**\begin{**aligned**}&\mathfrak{**VO**}\_{**B,**1}(\mathbf{**v**}\_1)|\_{\mathbf{**p**}\_0**,**\mathbf{**p**}\_{**B**0}**,**\mathbf{**v**}\_{**B**0}}\\&\mathfrak{**VO**}\_{**B,**2}(\mathbf{**p**}\_1(\mathbf{**v**}\_1)**,**\mathbf{**v**}\_2)|\_{\mathbf{**p**}\_{**B**1}**,**\mathbf{**v**}\_{**B**1}}\\&\mathfrak{**VO**}\_{**B,**3}(\mathbf{**p**}\_2(\mathbf{**v**}\_1**,**\mathbf{**v**}\_2)**,**\mathbf{**v**}\_3)|\_{\mathbf{**p**}\_{**B**2}**,**\mathbf{**v**}\_{**B**2}}\\&\vdots\\&\end{***aligned***}**