RMUA2021 技术方案

西安交通大学

笃行战队

2020-12

传感器布置方案以及部分目标检测效果在以下链接视频展示

https://v.youku.com/v_show/id_XNTAyMTQwMDE1Ng==.html

密码：xjtu1111

1.嵌入式系统结构

嵌入式系统结构由硬件系统结构、通信链路和传感器信息同步三部分组成。

1.1 硬件系统结构

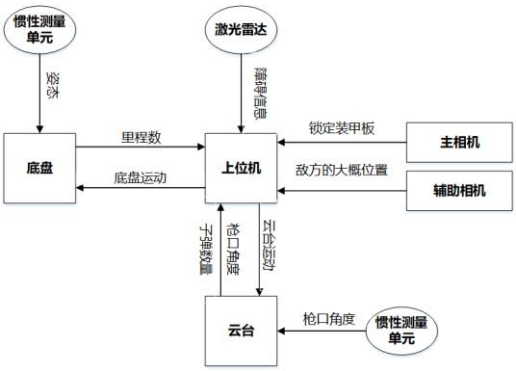


图 1-1 硬件系统结构

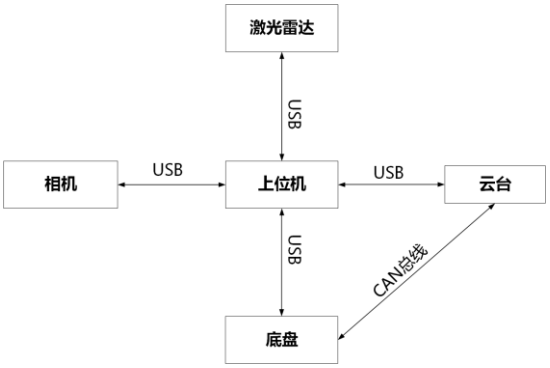


图 1-2 车辆通信框图

车辆数据的处理分为三个独立的部分:云台、底盘和上位机。上位机将相机采集到的信息进行处理，得到敌装甲位置，同时解算车辆的最佳攻击姿态，并传输给底盘主芯片和云台主芯片。底盘和云台控制器之间通过 CAN 总线通信。底盘主芯片和云台主芯片接收姿态信息之后，车辆将移动到最佳攻击姿势。同时，上位机还将从底盘和云台收集数据作为反馈，以此作为决策和导航操作的基础。

1.2 传感器底层架构

车辆装有多个传感器，最终将传感器获取的信息汇总到上位机进行处理。由于上位机需要对多个传感器的信息进行融合处理，因此在传感器信息中添加时间戳并校准车辆时间基准,保证传感器信息的时间同步。

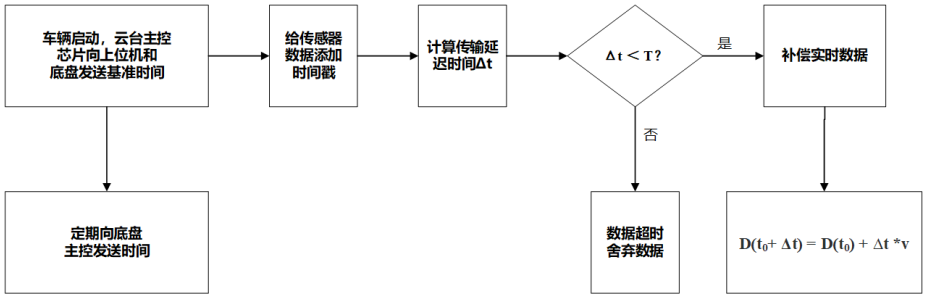


图 1-3 信息同步流程图

如果传输时间间隔小于设定阈值T,使用获得的数据和时间间隔 T 进行数据预测:假设时间传感器数据是$D(t_0)$，上位机接收到数据的时间间隔为Δt,数据变化速度为v,则可以根据公式 1-1 进行实时数据补偿，信息同步流程图如图 1-3 所示。

$$D(t_0 + \Delta t) = D(t_0) + \Delta t * v \qquad (1\text{-}1)$$

2.视觉感知

视觉感知模块的硬件由车载的深度相机和岗哨的工业相机组成，算法包括目标检测姿态估计、多相机协同的全局和局部目标定位跟踪预测、目标行动状态实时分析。

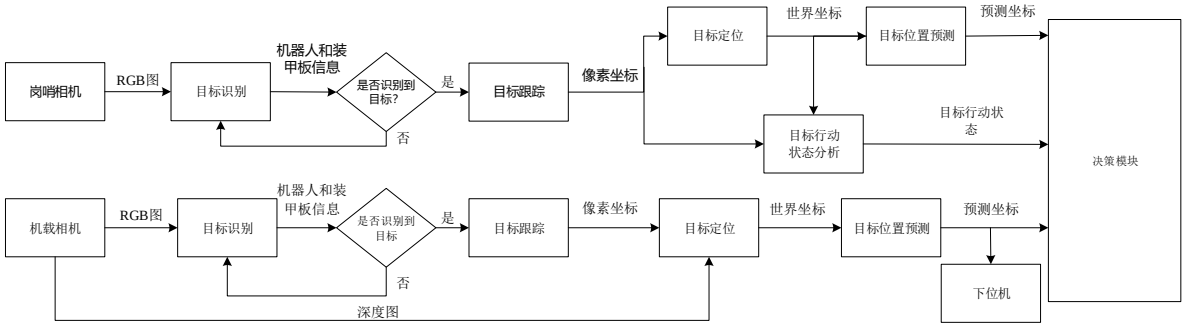


图 2-1　图像处理和视觉检测流程图

2.1 目标检测

2.1.1 岗哨视觉检测

哨岗视觉检测功能包括识别机器人和装甲板位置识别、机器人号码识别、机器人颜色识别。将识别的目标分为机器人（红蓝双方 1、2 号及阵亡机器人）和装甲板共 6 类进行检测。综合考虑精度和速度，我们采用 YOLOv5-l 网络结构实现哨岗目标检测。

2.1.2 机器人视觉检测

机器人车载视觉检测功能包括机器人装甲板识别、号码识别、颜色识别角度识别，从而提供敌方机器人相对我方机器人的精确位置信息。相比于岗哨，机器人上位机性能较为有限，因此我们采用了经过裁剪的 YOLOv5-m 网络结构。并用图 2-2 所示的基于泰勒展开的方法压缩模型。

由于不同部位装甲的伤害值不同，为了伤害最大化需要对目标姿态进行识别。我们在前述机器人分类的基础上加入角度特征重新划分类别，实现对机器人的角度识别，重新划分为 34 类（红蓝双方 1、2 号机器人的 8 种角度以及灰色机器人和装甲板）。

我们的视觉检测数据集构成如图 2-3 所示。我们增加了数据集中不同战队机器人外观的多样性，也融入了部分真实比赛的光线环境特征。同时增加数据集中战队的多样性，避免在训练过程中对本队机器人产生过拟合。

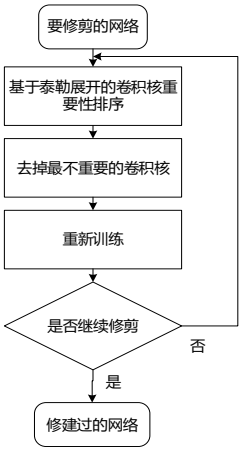


图 2-2 基于泰勒展开的型修剪流程图

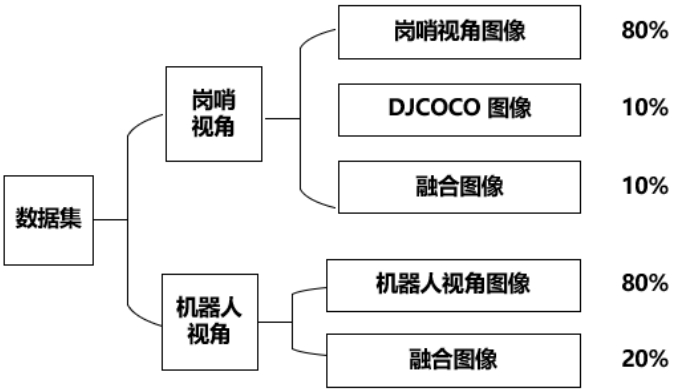


图 2-3 数据集构成

2.目标定位

2.1 岗哨相机位姿的获取

我们使用场地中的视觉标签来标定两个岗哨相机的外参，由于视觉标签在场地中的位置已知，可以容易地取得 4 个以上的 3D 点在 2D 图像上的匹配点。为了最大限度地利用场地视觉标签提升相机外参标定精度，我们采用 EPnP 算法对相机的位姿进行估计，然后根据最小化重投影误差的需求，构建最小二乘问题来对估计值进行非线性优化：

$$\boldsymbol{\xi}^* = argmin_{\xi} \frac{1}{2} \sum_{i=1}^{n} \left\| \boldsymbol{u_i} - \frac{1}{s_i} \boldsymbol{K} \exp(\boldsymbol{\xi}^\wedge) \boldsymbol{P}_i \right\| \tag{2-1}$$

2.2 岗哨的目标定位

选取装甲板中心作为特征点，由于它具有一个已知且固定的高度，如图 2-4 所示，通过计算相机光心 O和该点 A形成的直线 OA，与该点所在高度平面 H_A之间的交点位置，就可以确定该点在地面上的投影位置 A' 。

如图 2-5，根据视野中机器人能观察到的装甲板数量及目标机器人的尺寸特征，计算出目标机器人处于 R_1和 R_2两种可能的位姿时的中心位置 C_1和 C_2, 取 C_1C_2的中点 C_0作为目标位置的估计点。

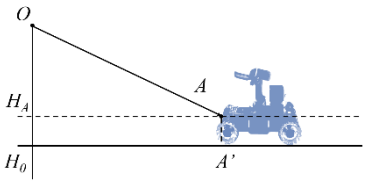


图 2-4 特征点位置估计

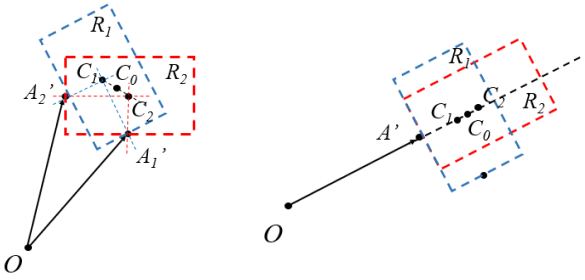


图 2-5 目标机器人位置估计

（a）检测到两块装甲板（b）只能检测到一块装甲板

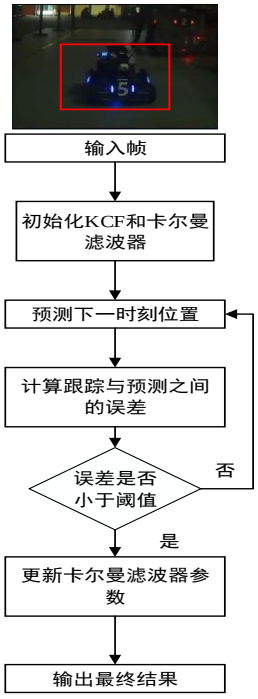


图 2-5 跟踪预测算法流程

2.3 岗哨的双相机协同

当距离较远时，测量会产生较大误差，因此在场地两侧靠近岗哨的区域，我们选择相信更近的相机得到的目标位置信息，而在场地中央的区域，我们将两相机得到的目标位置信息进行差分融合，得到更精确的位置观测信息：

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix} \begin{bmatrix} \frac{d_1}{d_1+d_2} \\ \frac{d_2}{d_1+d_2} \end{bmatrix} \tag{2-2}$$

2.4 目标跟踪与轨迹预测

为了精确打击地方机器人，需要对敌方车辆的运行轨迹进行跟踪预测。跟踪预测方法采用核相关滤波(KCF)跟踪与卡尔曼滤波(KF)相结合的方法，如图 2-5 所示，根据跟踪的结果作为输入不断更新敌方机器人的位置，再根据 KF 预测结果优化 KCF 的跟踪稳定性。KCF 采用了 HOG 特征，可以同时处理多通道数据，为多传感器数据融合提供可能。此外，KCF 通过岭回归训练目标检测器，使用循环矩阵扩大训练样本，简化了计算，在高帧率下仍有较高的计算效率和准确性。

2.5 目标行动状态分析

为了针对敌方机器人不同的行动状态采取不同的射击和运动策略，需要对其行动状态进行分析。我们将比赛中机器人的状态分为三类，普通运动、原地转动、原地左右摇摆。

当机器人在运动时，其检测框和装甲板框之间的相对位置、机器人的目标检测框位置在不同的行动状态下满足不同的模式，因此我们的解决方案如图 6 所示：检测机器人的目标框和其装甲板的中心点以及宽高信息，将机器人的目标框和其装甲板的目标框之间的相对位置、目标检测框的位置进行归一化，作为融合的位置信息向量输入 LSTM 中，实现对其行动状态的估计。

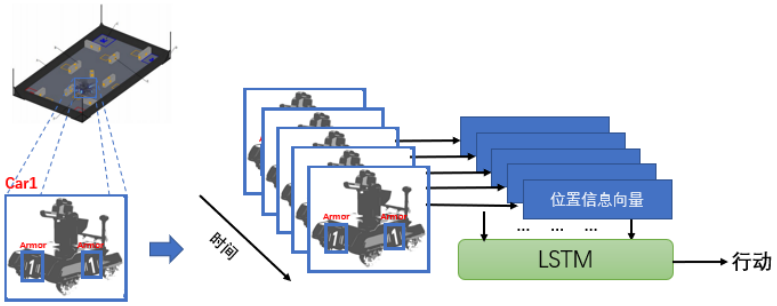


图 2-6　目标行动状态分析算法示意图

3、机器人定位

定位问题使用了改进后的 AMCL 算法，在原基于粒子滤波的 AMCL 算法的基础上，通过改进权值计算方式，加入图像量测信息进行多传感器信息融合，使用改进的非线性滤波器来提高定位精度。

3.1 改进 AMCL 权值计算方式

原有的 AMCL 算法中，在得到激光雷达获其他传感器的量测后，并没有使用这些量测来进行更新，而是直接使用这些真实量测与量测预测值的差别来求解粒子权值。我们借用 FASTSLAM2 的算法思想，在得到激光雷达和其他传感器量测后，首先进行更新，然后计算权值，概率方式表示的改进 AMCL 算法流程如下：

（1）一步预测。将第i个粒子的机器人位姿$[x,y,\varphi]^T$记为$x_{v,k|k-1}^i$，其方差记为$P_{v,k|k-1}^i$根据模型对机器人位姿进行一步预测。

$$x_{v,k|k-1}^i = f(x_{v,k-1}^i, u_k, 0) \tag{3-1}$$

（2）量测更新。对于每个粒子，使用最近一次对地标的观测对机器人位姿进行更新。将未进行量测更新的机器人位姿及其协方差记为$x_{0,v,k|k-1}^i$和$P_{0,v,k|k-1}^i$，假设观测并关联到的地标个数为M，对每个地标执行以下步骤：

a）量测预测。

$$\hat{z}_{k|k-1}^{i,j} = h(x_{j,v,k|k-1}^i, m_{j,k-1}^i) \tag{3-2}$$

b）得到雅各比矩阵。

$$H_{m_{k-1}}^{i,j} = \frac{\partial h(x_{j,v,k|k-1}^i, m_{j,k-1}^i)}{\partial m_{j,k-1}^i} \tag{3-3}$$

$$H_{x_{v,k|k-1}^i}^{i,j} = \frac{\partial h(x_{j,v,k|k-1}^i, m_j^i)}{\partial x_{j,v,k|k-1}^i} \tag{3-4}$$

c）计算信息协方差。

$$S_k^{i,j} = H_{m_{k-1}}^{i,j} P_{m_{k-1}}^{i,j} {H_{m_{k-1}}^{i,j}}^T + R$$

d）更新机器人位姿及其协方差

$$x_{j+1,v,k|k-1}^i = x_{j,v,k|k-1}^i + P_{j,v,k|k-1}^i \left(H_{x_{v,k|k-1}^i}^{i,j}\right)^T (S_k^{i,j})^{-1} (z_k^{i,j} - \hat{z}_{k|k-1}^{i,j}) \tag{3-5}$$

$$P_{j+1,v,k|k-1}^i = \left(\left(H_{x_{v,k|k-1}^i}^{i,j}\right)^T (S_k^{i,j})^{-1} H_{x_{v,k|k-1}^i}^{i,j} + (P_{j,v,k|k-1}^i)^{-1}\right)^{-1} \tag{3-6}$$

将经过所有观测并关联到的地标量测更新后的机器人位姿及其协方差记为$x_{M,v,k|k-1}^i$和$P_{M,v,k|k-1}^i$。

（3）权值计算。计算权值需要三个概率值。首先在$x_{v,k|k-1}^i \sim N(x_{M,v,k|k-1}^i, P_{M,v,k|k-1}^i)$中进行一次采样，采样值记为$x_{S,v,k|k-1}^i$并且

$$x_{v,k}^i = x_{S,v,k|k-1}^i$$

在高斯假设下，可以将其写为：

$$p(z_k|x_k,u_k) = \prod_{j=1}^{M} \frac{1}{\sqrt{2\pi}} e^{\frac{\left(z_k^{i,j} - \hat{z}_{S,k|k-1}^{i,j}\right)^2}{S_k^{i,j}}} \tag{3-7}$$

式中，$\hat{z}_{S,k|k-1}^{i,j} = h(x_{S,v,k|k-1}^{i}, m_{j,k-1}^{i})$。

$$p(x_k|x_{k-1},u_k) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\left(x_{S,v,k|k-1}^{i} - x_{0,v,k|k-1}^{i}\right)^2}{P_{0,v,k|k-1}^{i}}} \quad (3-8)$$

$$p(x_k|x_{k-1},z_k,u_k) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\left(x_{S,v,k|k-1}^{i} - x_{M,v,k|k-1}^{i}\right)^2}{P_{M,v,k|k-1}^{i}}} \quad (3-9)$$

$$w_k = w_{k-1} \frac{p(x_k|x_{k-1},u_k)p(z_k|x_k,u_k)}{p(x_k|x_{k-1},z_k,u_k)} \quad (3-10)$$

将式(3-7)到式(3-9)带入式(3-10)中，即获得权值。

（4） 增加新地标和进行重采样。

3.2 加入图像量测的多传感器信息融合

竞赛场的中的障碍物上会贴有独特的视觉标签。除过使用激光雷达进行定位外我们还使用机器人车载的视觉相机对视觉标签进行识别，获取机器人相对已知障碍物的坐标，同激光雷达数据进行融合对定位信息进行修正。

3.3 非线性滤波器 UCF

UCF 是基于线性最小均方差估计（LMMSE），使用非相关函数进行扩维从而达到更好的性能的滤波器。该算法将原始量测进行了变换，将变换后的量测扩维到原始量测中，使得量测从线性空间扩大到了非线性空间，使用扩维后的量测对状态进行更新，达到更好的滤波效果。利用非相关变换产生的确定性采样点计算x_k与扩维后量测$z_k^a = [z_k^T, y_k^T]^T$的互协方差$P_k^{xz^a}$以及z_k^a的协方差$P_k^{z^a}$，则该非线性滤波器的更新方程为：

$$\hat{x}_k = \hat{x}_{k|k-1} + P_k^{xz^a}\left(P_k^{z^a}\right)^{-1}\left(z_k^a - \left[\hat{z}_{k|k-1}^T, \hat{y}_k^T\right]^T\right) \quad (3-11)$$

$$P_k = P_{k|k-1} - P_k^{xz^a}\left(P_k^{z^a}\right)^{-1} P_k^{xz^a} \quad (3-12)$$

4、路径规划

4.1 软件架构

路径规划参考了 ROS 中的导航包。需要计算从里程计到坐标系的转换，从地图到里程计坐标系的转换。局部规划器基于全局规划器得出的全局路径进行规划以避障。软件架构如图 4-1 所示。

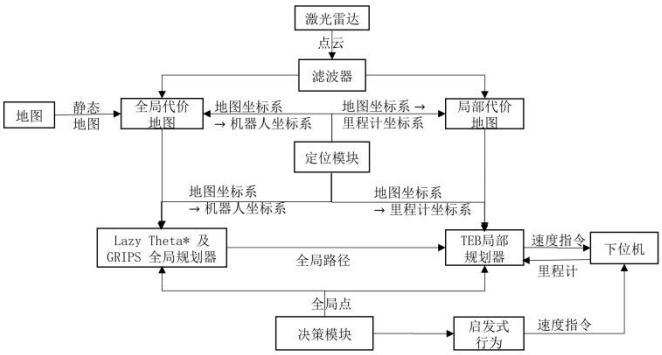


图 4-1. 路径规划软件架构

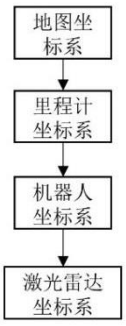


图 4-2. 坐标系转换

4.2 算法

4.2.1 建立分层的代价地图

分层代价地图基于语义信息实现实时更新，包括以下各层：(a) 静态地图层 (b) 警告区层 (c) 障碍层 (d) 预测层 (e) 膨胀层。在更新地图时，障碍层通过滤波后的激光雷达数据计算并更新障碍的边界，然后针对可能出现的其他机器人，基于其估计位置与速度进行预测，然后根据安全系数进行一定的膨胀处理，得到最终的边界。全局代价地图和局部代价地图分别包括不同的地图层。

4.2.2 基于 lazy theta*和 GRIPS 的全局规划器

基于 lazy theta*的全局规划器计算量小，可以向任意方向运动。GRIPS 可以对生成的全局路径

进行平滑处理。评价函数为：

$$f(n) = g(n) + w(n)h(n) \quad (4-1)$$

其中， n 是路径点的索引， g(n)是从起始点到当前点的代价， h(n)是当前点到终点的最小代价的启发式估计。w(n)是 h(n)的权重。初始时，w(n)设得较大，引导路径更快地向终点延伸；在搜索将结束时，寻找到到达终点的最优路径变得更加重要，因此 w(n) 可以设得较小。

首先，Lazy theta*计算当前点的相邻点的$f(n)$，并将当前点设为邻点的扩展节点，将当前点的父节点设为相邻点的父节点。然后将相邻节点添加到待扩展节点集合中，在上述集合中选择$f(n)$最小的点作为下一个扩展点。检测当前扩展点与其父节点之间是否有视线(line of sight, LOS)。如果没有，在要扩展的节点集合中，找到当前点的相邻扩展节点和当前点的父节点之间，并且距离父节点最近的一个点(P_{inside})。将当前点的父节点改为P_{inside}，更新当前点的$f(n)$和 $g(n)$，然后展开它的邻居。但如果存在视线，则直接扩展当前点的邻节点集。(图 4-3)

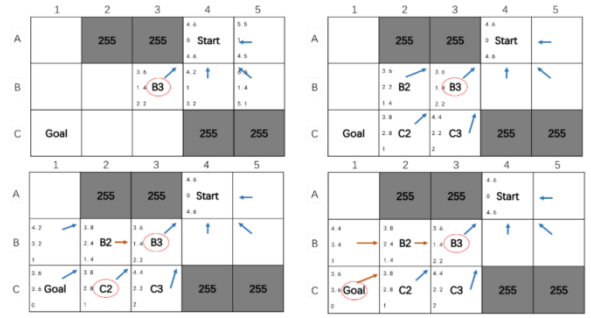
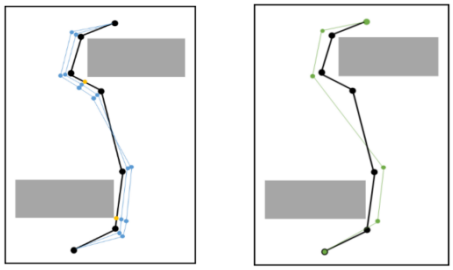


图 4-3. Lazy theta*生成路径的过程



(a) (b)

图 4-4. (a)基于梯度的路径变形 (b)最小代价路径

生成的路径可能不够平滑，或者距离障碍物很近，不能直接使用，需要通过 GRIPS 来平滑路径。首先，插入一些距离障碍物最近的点，基于关键点(p)和障碍物之间的最小距离的梯度($\nabla D[p]$)来修正路径。然后确定不可移动点：若连接一个点的两个相邻点的线通过障碍，则这个点是不可移动的。通过确定相邻不可移动点之间的最佳路径获得最小代价路径(图 4-4)。

$$p = p - \eta \frac{\nabla D[p]}{D[p]} (\eta \in (0,1)) \quad (4-2)$$

4.2.3 基于 TEB 的局部规划器

局部规划器通过 TEB 实现速度控制和避障。通过调整速度、运动学和间隔时间等约束条件的权重，可以解决针对不同环境和机器人的优化规划问题。优化问题如下：

$$\min_{B} \sum_{k=1}^{n-1} \Delta T_k^2 \quad (4-3)$$

其中：

$$\mathbf{s}_1 = \mathbf{s}_c, \quad \mathbf{s}_n = \mathbf{s}_f, \quad 0 \le \Delta T_k \le \Delta T_{\max} \quad \mathbf{h}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) = \mathbf{0}, \quad \tilde{r}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) \ge 0$$

$$\mathbf{o}_k(\mathbf{s}_k) \ge 0, \quad \boldsymbol{v}_k(\mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_k) \ge 0, (k = 1,2,\dots,n-1)$$

$$\alpha_k(\mathbf{s}_{k+2}, \mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_{k+1}, \Delta T_k) \ge 0, (k = 2,3,\dots,n-2)$$

$$\alpha_1(\mathbf{s}_2, \mathbf{s}_1, \Delta T_1) \ge 0, \quad \alpha_n(\mathbf{s}_n, \mathbf{s}_{n-1}, \Delta T_{n-1}) \ge 0$$

上述各式的具体含义可参考。该优化问题是 NLP 问题，需要转化成一个近似的非线性二次优化问题。

通过 g2o 解优化问题得到最优路径 B *，下发到 STM32 的指令为：

$$\boldsymbol{v}_i = \begin{bmatrix} \cos(\beta_{i+1} - \beta_i) & \sin(\beta_{i+1} - \beta_i) & 0 \\ -\sin(\beta_{i+1} - \beta_i) & \cos(\beta_{i+1} - \beta_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (x_{i+1} - x_i)/\Delta T_i \\ (y_{i+1} - y_i)/\Delta T_i \\ (\beta_{i+1} - \beta_i)/\Delta T_i \end{bmatrix} \quad (4-4)$$

$$\boldsymbol{a}_i = \frac{2(\boldsymbol{v}_{i+1} - \boldsymbol{v}_i)}{\Delta T_i + \Delta T_{i+1}} \quad (4-5)$$

5、机器人决策部分

5.1 总体决策框架

在决策部分，我们将裁判系统信息和机器人通过传感器融合或滤波得到的信息作为输入，输出是机器人的具体或者抽象的行为动作。多机器人集成决策系统是在行为树的基础下，结合了先验知识对部分控制节点进行深度强化学习的算法框架。在任务调度层面，我们选取了一些传统方法无法很好实现的动作，如目标跟踪和躲避目标，通过深度强化学习对目标动作进行训练的端到端算法。并采用竞争机制，在仿真环境中实现了跟踪与逃跑的对抗性游戏训练。具体决策部分的系统框架如下：

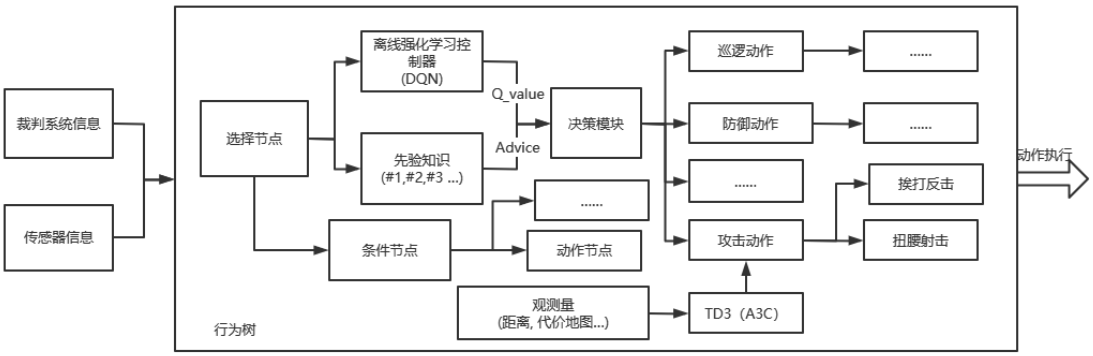


图 5-1 决策部分的系统框图

5.2 多机器人集成决策系统

5.2.1 算法框架

决策系统以行为树（Behavior Tree）为总体框架，行为树作为状态机的升级版，其结构更加鲜明，易于管理和实现。但行为树只是即时的逻辑判断，不考虑过去和动作后的状况，且部分条件过于模糊化。所以在个别控制分支处，采用结合先验知识的深度 Q 神经网络算法。该算法在充分利用观测信息的基础上，利用行为树中已有的先验知识干预 DQN 算法训练中的动作选择过程，从而降低动作选择的随机性，加快收敛速度。

先验知识是先于经验的知识，在分类问题中，我们可以基于某些特征进行预分类。比如：比赛初始，视野无敌人血量弹药充裕，执行巡逻任务；视野内出现敌人且我方血量和弹药状态压制，进行追击等，避免了初期随机性较大产生的无用探索，让机器人学习“规则”。

PK-DQN 中引入了特征状态序列$(P = [p_1\ p_2 \dots p_i \dots p_n],\ i = 1,2,3 \dots)$，结合 DQN 的 Q 值得到:

$$Q_f = \mu Q_p + (1-\mu)Q,\ 0 < \mu \le 1 \qquad (5\text{-}1)$$

其中 Q 为 DQN 算法根据当前状态估计出的动作选择 Q 值向量，Q_p为先验 Q 值向量，μ值表示特征状态与选择动作的相关性。当出现特征状态p^*时，根据式()计算出每个动作的Q_f值，直接根据最大Q_f值选择动作；当未出现特征状态p_i时，根据 DQN 算法估计动作的 Q 值，然后根据 ε-greedy 规则选择动作。算法流程图如下：

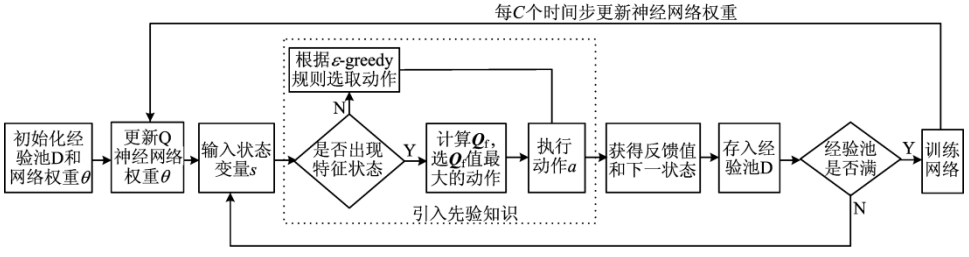


图 5-2 PK-DQN 算法流程

5.2.2 算法实现

在具体算法实现中，我们根据行为树的控制节点分支条件作为先验知识，我们先简单把机器人行为大体分为 Patrol, Attack, Defense, Support 四个类型作为输出，两个个体就有4^2种动作组合，以相关性较强的信息作为观测输入值（如各机器人血量、剩余子弹数、Buff 状态、比赛信息等），结合比赛规则和收敛性确定奖励值（Reward），如单位时间净胜血量，Buff 的增益或减损，是否获取敌方坐标位置等。以一定游戏周期作为一个 step，一局游戏作为一个 episode，在仿真环境中不断训练更新神经网络的参数。经过一定训练后，在实际环境中测试并训练，对比效果。

5.3 单机器人任务调度

5.3.1 算法结构

在单机器人任务调度层面，本方案设计了基于强化学习对目标跟踪和躲避动作进行端到端的训练。与传统的将目标识别和底层控制分离为两个独立任务的方法相比，我们直接以规划模块构建的代价地图以及测量到的目标距离作为输入特征，输出为底盘运动矢量。利用对抗性博弈的思想，在仿真环境中同时训练跟踪和逃逸网络，促进和提高神经网络的训练效率。

5.3.2 算法实现

我们的跟踪和逃逸使用同一套网络结构，主要由三部分组成，如下所示。特征编码器使用卷积层和完全连接层将输入成本映射编码为特征向量。我们构造了一个 LSTM 网络，融合了历史输入特征，得到了包含目标速度和运动方向等时间序列特征的序列编码器表示。

强化学习算法模块接收序列编码器的特征作为输入，并输出连续的控制量。本方案采用 TD3 算法消除了由于高估值函数而导致的次优策略更新和行为偏差，并在 DDPG 算法的基础上提出了消减双 Q 学习来减小方差。

$$r_1 = -\left(\frac{(distance_{agent-enemy1})}{arena.length+arena.width} + \lambda_1|\omega_1|\right) \tag{5-2}$$

$$r_2 = -\left(\frac{distance_{agent-enemy2}}{attackRange} + \lambda_2|\omega_2|\right) \tag{5-3}$$

在强化学习中，奖励函数指导着 agent 的学习，对最终的学习效果起着至关重要的作用。对于比赛场景的跟踪和逃逸任务，我们期望跟踪器在跟踪 enemy1 时避开 enemy2，因此我们根据机器人之间的距离设计了奖赏函数。

r_1被设计成跟踪敌人的奖励，竞技场是比赛场地，ω是角偏差。r_2被设计成接近敌人的惩罚。我们还根据障碍物或禁区设置额外的奖励r_{ex}作为惩罚。所以我们的总回报函数是$r = r_1 + r_2 + r_{ex}$。

6、基于不完全微分 PID 的云台控制

6.1 控制方案概述

云台是一二自由度旋转机构，控制目标是快速、准确地达到预定的位置。在 PID 算法中加入一阶惯性环节，使用不完全微分 PID 达到目标；同时考虑子弹重力和空气阻力对弹道的影响，建立射击模型，对 pitch 轴进行补偿。云台运动控制方案如图 6-1 所示，该方法提高了系统的稳定性和射击准确度。

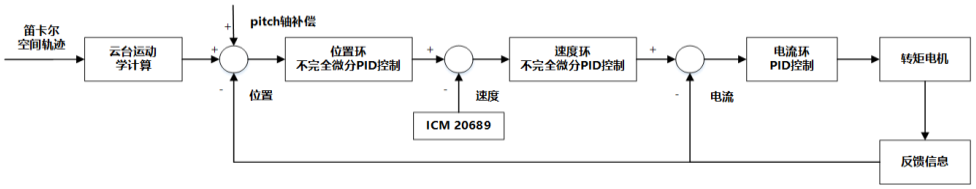


图 6-1 控制方案

6.2 射击模型分析及俯仰轴角度补偿

子弹在飞行中受空气阻力影响会击中较射出点更低的位置。因此，考虑重力和水平方向的空气阻力，建立图 6-2 中的射击模型如下:

设目标点为(x,y)，子弹射出速度为v_0，θ为 pitch 轴角度。在x方向上的空气阻力计算公式为:

$$f = C\rho S{v_x}^2/2 \quad (6\text{-}1)$$

设$k_0 = C\rho S/2$，则$f = k_0{v_x}^2$。由牛顿第二定律:

$$-f_x/m = a = dv_x/dt \quad (6\text{-}2)$$

得

$v_x = v_{x0}/(k_1 v_{x0} t + 1)$,其中$k_1 = k_0/m$,所以水平方向的位移公式为：$x = \int_0^t v_x dt = (ln(k_1 v_{x0} t + 1))/k_1$。

x已知，k_1由场馆的环境和子弹的物理特性确定，$v_{x0} = v_0 cos\theta$，θ为俯仰角，所以时间t可求，将t代入$y' = gt^2/2 + v_0 sin\theta$，可以得到实际位置$y'$。

θ为一个变量,我们把$|y - y'| \le \varepsilon$设置为迭代终止条件，其中ε为给定的精度。

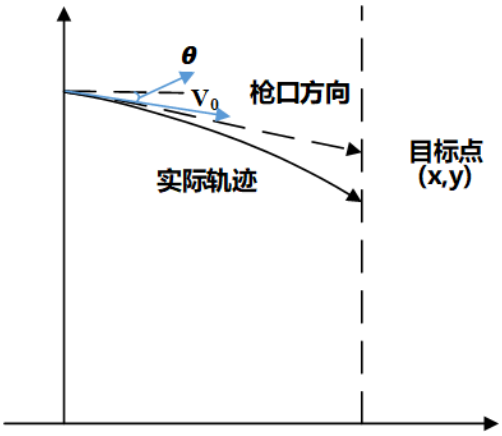


图 6-2 射击模型

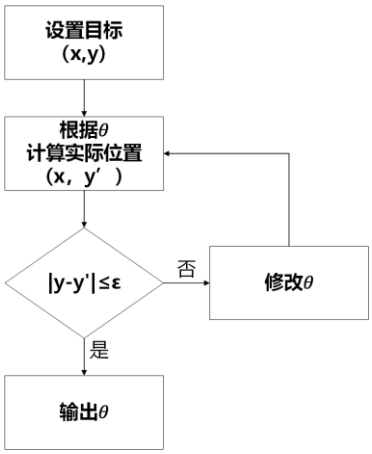


图 6-3 求解流程图

当给定一个目标点(x,y)、v_0和以忽略空气阻力的斜抛运动模型计算得初始的θ，通过上面的迭代法，可以获得 pitch 轴角补偿值以提高射击精度，求解流程图如图 6-3。

6.3 底盘云台控制模式

将机器人的运动分为搜索和交战两个状态。与有人遥控的机器人不同，自动机器人并不需要考虑运动过程中底盘与云台不跟随可能带来的运动偏差。搜索状态下，底盘根据上位机发送的运动指令经运动学结算后进行运动，此时云台 YAW 轴方向摆动进行索敌。在交战状态下，云台根据视觉识别结果对敌射击，底盘以云台朝向为中心，以云台 YAW 为旋转中心左右旋转以规避敌方射击。

7、通信链路

岗哨可以获得较单个机器人更为丰富的信息，合理的进行数据处理并进行两台机器人间的协同策

略控制可以有效提高索敌效率。我们利用 WIFI 进行机器人与岗哨间的双向通信。岗哨作为服务器向作为客户端的机器人发送包括敌方坐标、运动状态在内的竞赛信息。

8、扩展目标跟踪

使用视觉方案进行目标识别时常将目标简化为点目标。为更丰富地获取目标信息，我们结合 Mid-70 激光雷达同视觉传感器进行配准后可获得包括目标距离、底盘朝向、枪口朝向、运动方向的更多信息。

9、设备选型

竞赛规则规定参赛队伍使用统一的移动机器人平台。需要结合技术需要选择传感器及计算平台。将竞赛所需设备分为岗哨和机器人两部分。岗哨的作用是在竞赛期间提供全局的目标信息为决策提供依据。由于岗哨传感器安装位置距离战场目标较远，传感器需要具有较大的量程和分辨率。岗哨上使用单目工业相机配合长焦镜头作为视觉传感器。为了提高岗哨对于远距离目标的观测精度，除过视觉相机我们在岗哨上安装 LIVOX Mid-70 激光雷达。相比单纯的视觉方案，我们将激光雷达的点云信息和相机的图像进行配准后可以更精准的获取目标的方位距离等信息。

机器人车载设备主要包括视觉传感器、定位用激光雷达和计算平台。由于机器人距离目标较近，目标在视野中运动较快需要选择视角大帧率高的视觉相机。同时为了实现远程射击，需要有较长焦距我们使用单目高帧率工业相机结合中长焦镜头。3 米以内近程目标我们使用深度相机作为传感器。同时使用 LIVOX Mid-70 激光雷达进行目标扩展。在定位方面我们选用 RPLIDAR A2 作为定位传感器，为避免机器人云台的遮挡，获取机器人周围数据信息，我们将激光雷达反置在底盘下方。由于岗哨和机器人都需要进行大量的图像和深度学习运算，我们选择 DJI-Manifold2-G 作为计算平台。主要设备型号及参数如表所示。

传感器	型号	参数
单线雷达	Rplidar A2	360°，8000 点/s，16m
固态雷达	Mid-70	70.4°，100000 点/s，90m
工业相机	Daheng MER-239-168U3C	168fps，1920*1200
深度相机	RealSense D435i	90fps，1280*720
计算平台	Manifold2-G	TX2, 8GB RAM

表 9-1 主要设备型号及参数