如图所示对于一个platoon，其主要包含四个元素node dynamics (ND), information flow network (IFN), distributed controller (DC), formation geometry (FG)。

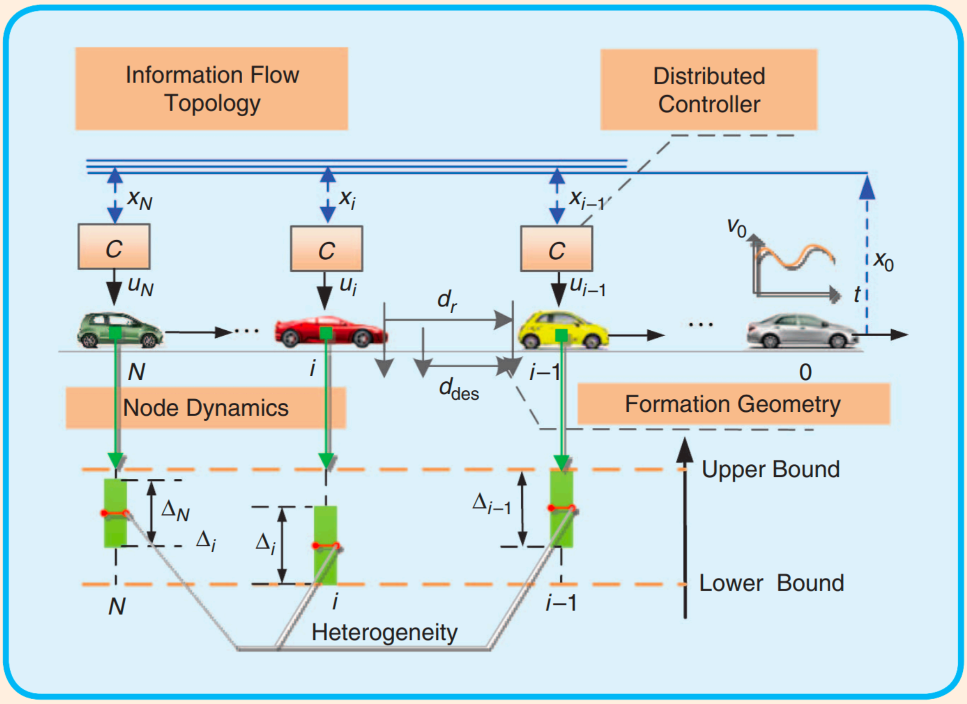


FIG 1 Four major components of a platoon [22], [23]: 1) node dynamics, 2) information flow network, 3) distributed controller, 4) geometry formation; where d_r is the actual relative distance, d_{des} is the desired distance, u_i is the the control signal, x_i is the state, Δ_i denote the dynamical uncertainty, and C denotes the controller.

Node dynamics

third-order相比second-order的好处是使得state增加了一维，可以更好的模拟powertrain dynamics的输入输出。

Single integrator model

这是一种最简单的情况，直接将车辆的速度作为control input，将位置作为state

$$\dot{p}_i(t) = u_i(t)$$

Second-order linear model (position-velocity)

例如：

$$\begin{cases} \dot{p}_i = v_i \\ \dot{v}_i = u_i, \quad i = 0, 1, \cdots, n \end{cases}$$

$$u_i = f_i\left(p_{i-1}, p_i, p_{i+1}, p_{s_i}\right) + g_i\left(v_{i-1}, v_i, v_{i+1}\right), \quad i = 1, \ldots, n-1$$

其中u_i为车的加速度，$f_i(*)$和$g_i(*)$分别为位置和速度的函数。

Second-order nonlinear model (position-velocity)

例如：

$$
\begin{aligned}
&\dot{p}_i(t)=v_i(t), i \in \mathcal{V}_N \\
&\dot{v}_i(t)=\operatorname{sat}\left(u_i(t)\right)+f_i\left(p_i(t), v_i(t), t\right)+w_i(t)
\end{aligned}
$$

其中$f_i(*)$为环境的非线性干扰，例如随机的加速干扰、风阻、路况等等。$\|w_i(t)\| \leq \bar{w}$为控制输入的干扰。$sat(*)$函数规定了$u_i(t)$的上界为u_{Mi}。同样，其为一种开环控制。

$$
\operatorname{sat}\left(u_i(t)\right)= \begin{cases}u_{Mi} \operatorname{sgn}\left(u_i(t)\right), & \text { if }\left|u_i(t)\right| \geq u_{Mi} \\ u_i(t), & \text { if }\left|u_i(t)\right| \leq u_{Mi}\end{cases}
$$

Third-order nonlinear model (position-velocity-force)

形如：

$$
\begin{aligned}
&\dot{p}_i(t)=v_i(t) \\
&\dot{v}_i(t)=\tfrac{1}{m_i} F_i(t)-\tfrac{K_{di}}{m_i} v_i^2(t)-\tfrac{d_{mi}}{m_i} \\
&\dot{F}_i(t)=\tfrac{1}{\tau_i} u_i(t)-\tfrac{1}{\tau_i} F_i(t)
\end{aligned}
$$

其中$F_i(t)$为第i辆车的引擎施加在车上的力，m_i为第i辆车的质量，τ_i为第i辆车的引擎时延，K_{di}为气动阻力参数，d_{mi}为机器阻力。

Third-order nonlinear model (position-velocity-acceleration)

形如：

$$
\begin{aligned}
\dot{p}_i(t) &= v_i(t) \\
\dot{v}_i(t) &= a_i(t) \\
\dot{a}_i(t) &= \frac{1}{m_i \tau_i} u_i(t)-\frac{2 K_{di}}{m_i} v_i(t) a_i(t)-\frac{1}{\tau_i}\left(a_i(t)+\frac{K_{di}}{m_i} v_i^2(t)+\frac{d_{mi}}{m_i}\right)
\end{aligned}
$$

和position-velocity-force没什么区别，将$F_i(t)$用$a_i(t)$和$v_i(t)$替代即可，这样的好处是加速度更容易分析。

Third-order linear model (position-velocity-acceleration)

形如：

$$
\begin{aligned}
&\dot{p}_i(t)=v_i(t) \\
&\dot{v}_i(t)=a_i(t) \\
&\dot{a}_i(t)=\frac{1}{\tau_i} c_i(t)-\frac{1}{\tau_i} a_i(t)
\end{aligned}
$$

其中state包含位置、速度、加速度。其中控制输入与加速度的微分成正比。

当控制器已知环境参数时，可将非线性模型线性化，$u_i(t)$满足：

$$u_i(t) = m_i c_i(t) + K_{di} v_i^2(t) + d_{mi} + 2\tau_i K_{di} v_i(t) a_i(t)$$

Information Flow Topology (IFT)

一些常用的Information Flow传递方式如下，(a)和(c)图分别为predecessor following (PF)、bidirectional (BD) topologies，它们常用于使用雷达等传感器的车队中，只能测量相邻车辆的速度、位置等信息。(b)、(d)、(e)、(f)是当引入车联网后的车车通信模型，分别表示predecessor-following leader (PFL) type, bidirectional leader (BDL) type, two predecessorfollowing (TPF) type, and two predecessor-following leader (TPFL) type。

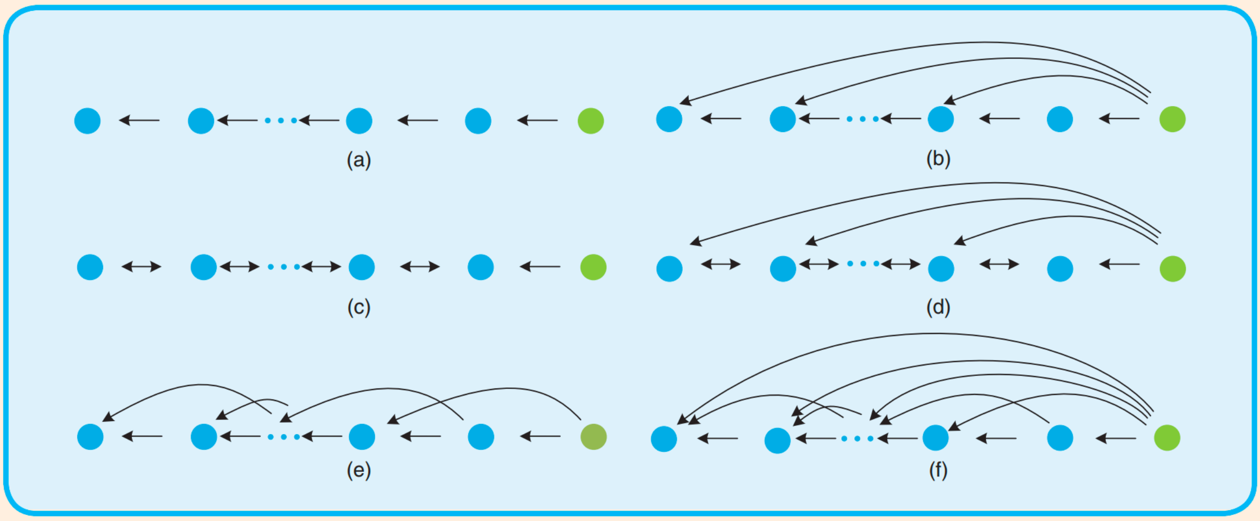


FIG 2 Typical IFTs: (a) PF, (b) BD, (c) PFL, (d) BDL, (e) TPF, (f) TPFL.

Distributed Controller

四种常用的distributed controller分别为linear consensus control, robust control, distributed sliding mode control, distributed model predictive control。前两者用于linear dynamic model，后两者用于non-linear dynamic model。

linear consensus control

线性控制的通常形式为：

$$\begin{aligned} u_i(t) = -\sum_{j \in \mathbb{I}_i} & [k_{ij,p} \left(p_i \left(t - \gamma_{ii}\right) - p_j \left(t - \gamma_{ij}\right) - d_{i,j}\right) \\ & + k_{ij,v} \left(v_i \left(t - \gamma_{ii}\right) - v_j \left(t - \gamma_{ij}\right)\right) \\ & + k_{ij,a} \left(a_i \left(t - \gamma_{ij}\right) - a_j \left(t - \gamma_{ij}\right)\right)] \end{aligned}$$

其中$k_{ij,\#}(\# = p, v, a)$，其代表controller gain，分别是间距误差的参数、速度误差的参数和加速度误差的参数。

γ_{ii}为车辆接受自身state的时延，γ_{ij}为i车辆接受j车辆state的时延。

distributed robust control

H_∞ control可以保证string stability，并且适合具有异构车辆、具有不确定的dynamics和时延的车队。但是缺点是它只适合于特定的车队，且当车队的通信模式变化，控制模型要重新设计。

distributed sliding control

$\dot{x} = f(x) + u$，控制目标为$x- > x_d$，其中$\|f(x)\| < \rho(x)$

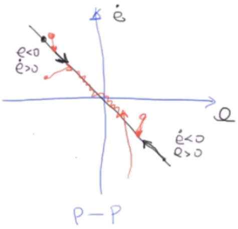
$\dot{e} = \dot{x}_d - \dot{x} = \dot{x}_d - f(x) - u$

令$u = \dot{x}_d + ke + \rho * sgn(e)$

这样就可以令$lim_\infty e = x_d - x = 0$当t趋向于无穷时。

代入得$\dot{e} = -ke - f(x) - \rho(x)sgn(e)$

其中后面的部分$-f(x) - \rho(x)sgn(e)$为控制项，其会不断的将偏离的点滑到这条黑线$\dot{e} = -ke$上。



distributed model predictive control

通过模型来预测系统在某一位来实际内的表现来进行优化控制。

有系统：$x_{k+1} = Ax_k + Bu_k$

step 1:估计/测量系统当前的状态。

step 2:基于$u_k, u_{k+1} \ldots u_{k+n}$来进行最优化。（预测区间/控制区间）

代价函数$J = \sum_k^{N-1} E_k^T Q E_k + u_k^T R u_k + E_N^T F E_N$

三项分别为每个step的error误差、控制的能量消耗、最终时刻的误差。

step 3:只施加u_k

重点在于如何做最优化，最常用的方法为二次规划QP。

其一般形式为：$min\ Z^T Q Z + C^T Z$

假设在k时刻，预测区间中的状态和控制输入可以表达为：

在 k 时刻

$$X_k = \begin{bmatrix} x(k|k) \\ x(k+1|k) \\ x(k+2|k) \\ \vdots \\ x(k+N|k) \end{bmatrix} \qquad U_k = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+i|k) \\ u(k+N-1|k) \end{bmatrix}$$

在k时刻预测

预测区间 Predictive Horizon.

代价函数为：

Cost function 代价

$$J = \sum_{i=0}^{N-1} \left(\underbrace{x(k+i|k)^T Q\, x(k+i|k)}_{\text{误差加权和}} + \underbrace{u(k+i|k)^T R\, u(k+i|k)}_{\text{输入加权和}} \right) + \underbrace{x(k+N)^T F\, x(k+N)}_{\text{终端误差}}$$

带如初始条件进行迭代：

$$X_k = \begin{bmatrix} x(k|k) = \overset{\text{初始条件}}{x_k} \\ x(k+1|k) = A x(k|k) + Bu(k|k) = Ax_k + Bu(k|k) \\ x(k+2|k) = A x(k+1|k) + Bu(k+1|k) = A^2 x_k + ABu(k|k) + Bu(k+1|k) \\ \vdots \\ x(k+N|k) = A^N x_k + A^{N-1}Bu(k|k) + \cdots + Bu(k+N-1|k) \end{bmatrix}$$

$$X_k = \begin{bmatrix} I \\ A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} x_k + \begin{bmatrix} 0 & 0 & \cdots & 0 \\ B & & & \\ AB & B & \ddots & \\ \vdots & & & B \\ A^{N-1}B & A^{N-2}B & & \end{bmatrix} \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix}$$

最后可以简化为：

$$\boxed{X_k = M x_k + C U_k}$$

将代价函数写成矩阵形式：

$$\boxed{X_k = M x_k + C U_k}$$

$$J = \sum_{i=0}^{N-1} \left(x(k+i|k)^T Q\, x(k+i|k) + \boxed{u(k+i|k)\, R\, u(k+i|k)} + x(k+N)^T F\, x(k+N) \right) = U_k^T \bar{R} U_k$$

$$x(k|k)^T Q\, x(k|k) + x(k+1|k)^T Q\, x(k+1|k) + \cdots + x(k+N-1|k)^T Q\, x(k+N-1|k) + x(k+N)^T F\, x(k+N)$$

$$= \begin{bmatrix} x(k|k) \\ x(k+1|k) \\ \vdots \\ x(k+N|k) \end{bmatrix}^T \underbrace{\begin{bmatrix} Q & & & \\ & Q & & \\ & & Q & \\ & & & \ddots \\ & & & & F \end{bmatrix}}_{\bar{Q}} \begin{bmatrix} x(k|k) \\ \vdots \\ x(k+N|k) \end{bmatrix} = X_k^T \bar{Q} X_k$$

$$J = X_k^T \bar{Q} X_k + U_k^T \bar{R} U_k$$

$$= (M x_k + C U_k)^T \bar{Q} (M x_k + C U_k) + U_k^T \bar{R} U_k$$

展开后化简，成为了二次规划的一般形式（一个二次项、一个线性项和一个初始项）：

$$= x_k^T \underbrace{M^T \bar{Q} M}_{G} x_k + \boxed{x_k^T M^T \bar{Q} C U_k}^T + \boxed{U_k^T C^T \bar{Q} M x_k} + U_k^T C^T \bar{Q} C U_k + U_k^T \bar{R} U_k$$

$$C^T \bar{Q} C + R = H$$

$$2 x_k^T \underbrace{C^T \bar{Q} M}_{E} U_k$$

$$\boxed{J = x_k^T G x_k + 2 x_k^T E U_k + U_k^T H U_k}$$

Formation Geometry (FG)

可以理解为控制的目标，它通常包括以下三种：1) constant distance (CD) policy, 2) constant time headway (CTH) policy, and 3) nonlinear distance (NLD) policy。

control objective：

a) to ensure all the vehicles in the same group to move at the same speed with the leader

b) to maintain the desired spaces between adjacent vehicles

constant distance policy

相邻两辆车的理想间距为常数，与速度无关（高交通容量）：

$$d_{i-1,i} = d_0, i \in \mathcal{N}$$

constant time headway policy

此时相邻两辆车的理想间距与速度有关（限制部分交通容量），其中t_h为time headway。

$$d_{i-1,i} = t_h v_i + d_0, i \in \mathcal{N}$$

nonlinear distance policy

这种情况下，两辆车的desired distance为关于速度的函数。这种方法有潜力在保证交通流稳定性的情况下提高交通容量。

$$d_{i-1,i} = g(v_i), \quad i \in \mathcal{N}$$