引入“scale”的说明：

在 control_auto 函数中，引入“scale”。

Scale =_params.pos_p .edivide(_parems.vel_max)

然后将所有的与位置有关的变量都乘以 scale，换算到 scaled space。这样处理的目的是将所有的位置信息都换算到同一个空间下，在这个空间里，位置的差值 position error = 1 的话，就会产生所期望的最大的“巡航速度”。

在 control_auto 中共 2 处使用 scale 的，但具体的使用目的不一样，下面进行进一步说明：

①在 cross_sphere_line 函数中使用，具体可以参考 cross_sphere_line()函数说明；

```
1081        /* move setpoint not faster than max allowed speed */
1082        math::Vector<3> pos_sp_old_s = _pos_sp.emult(scale);
1083
1084        /* difference between current and desired position setpoints, 1 = max speed */
1085        math::Vector<3> d_pos_m = (pos_sp_s - pos_sp_old_s).edivide(_params.pos_p);
1086        float d_pos_m_len = d_pos_m.length();
1087
1088        if (d_pos_m_len > dt) {
1089            pos_sp_s = pos_sp_old_s + (d_pos_m / d_pos_m_len * dt).emult(_params.pos_p);
1090        }
1091
1092        /* scale result back to normal space */
1093         pos_sp = pos_sp_s.edivide(scale);
```

②在此处(上图)使用：这里的使用目的是保证 position_setpoint 的变化率 sp_move_rate 不大于 “巡航速度”。

实际上就是**(pos_sp_s – pos_sp_old_s) / dt = sp_move_rate_s;**

sp_move_rate_s / _params.pos_p = position_error <1;

这部分代码当 previous.valid = false&&current.valid = true 情况下仍然执行，比如在 RTL 的 DESCEN STATE 下。

Cross_sphere_line()函数说明：

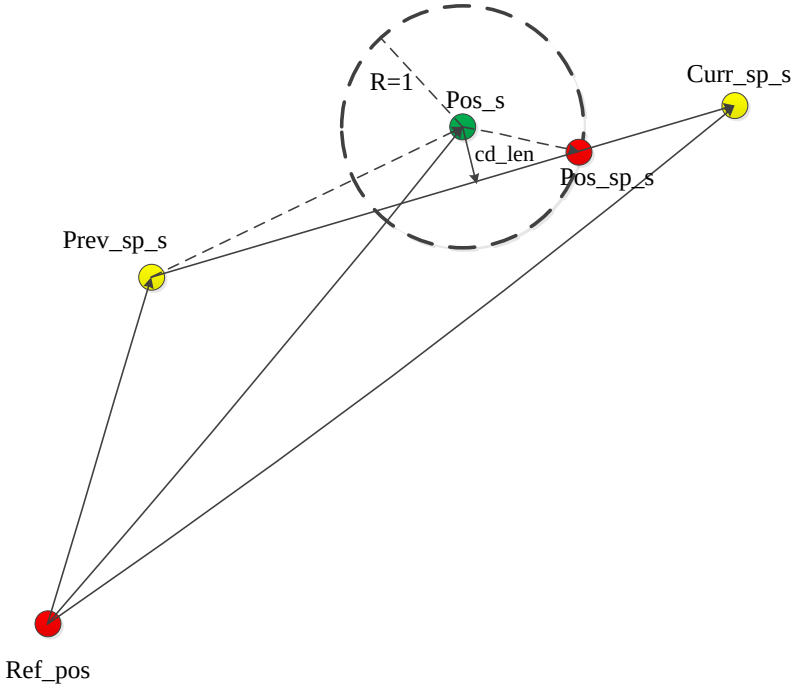
该函数在 control_auto 中调用，其作用为使飞机沿“previous_setpoint——current_setpoint trajectory”飞行。根据当前位置 postion 与 trajectory 的垂直距离 cd_len,来实时的改变 pos_sp。

如下图所示.

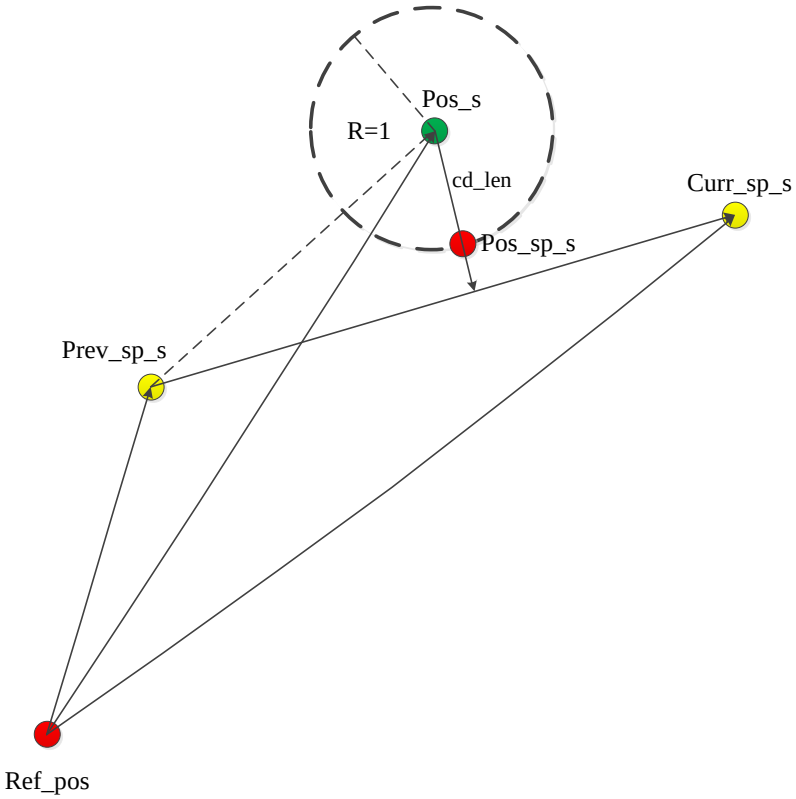
但值得注意的是:

①实际上是一个三维的球体(sphere),而图示的是一个二维的表示；

②球的半径选择为 r=1 的原因，是由于在 control_auto 中引入了“scale”的概念，半径为 1 的目的实际上就是相当于使期望速度达到“巡航速度”。



cd_len < sphere_r



cd_len > sphere_r