// PSEUDO-CODE for MECANUM or OMNI,
// with or without FIELD-CENTRIC CONTROL (gyro)

// First define your driver interface:

// For a 3-axis joystick (Z axis is twist), do this:

```
forward   = -Y;  // push joystick forward to go forward
right     =  X;  // push joystick to the right to strafe right
clockwise =  Z;  // twist joystick clockwise turn clockwise
```

// ... or for two 2-axis joysticks do this (Halo):

```
forward   = -Y1;  // push joystick1 forward to go forward
right     =  X1;  // push joystick1 to the right to strafe right
clockwise =  X2;  // push joystick2 to the right to rotate clockwise
```

// or this ("tank drive" interface plus strafe):

```
forward   = -(Y1+Y2)/2;  // push both joysticks forward to go forward
right     =  X2;         // push joystick2 to the right to strafe right
clockwise = -(Y1-Y2)/2;  // push joystick1 forward and pull joystick2 backward
                         // to rotate clockwise
```

// ... or for a single 2-axis joystick:

```
forward    = -Y;
right      =  (button1)? X:0;
clockwise  =  (button1)? 0:X;
```

// note: the above can only do 2 degrees of freedom at a time.
// Button1 selects Halo or Arcade.

// ... or another way to do a single 2-axis joystick:

```
forward    = -Y;
right      =  X;
clockwise  =  {-1 if(button2); +1 if(button4); else 0;}
```

// use button2 & button4 to "bump" rotate clockwise & counterclockwise.

// ... or any other driver interface scheme you like !

```
// Now add a tuning constant K for the "rotate" axis sensitivity.
// Start with K=0, and increase it very slowly (do not exceed K=1)
// to find the right value after you've got fwd/rev and strafe working:

clockwise = K*clockwise;


// OPTIONAL.  If desired, use the gyro angle "theta" for field-centric control:

// if "theta" is measured CLOCKWISE from the zero reference:

temp    =  forward*cos(theta) + right*sin(theta);
right   = -forward*sin(theta) + right*cos(theta);
forward =  temp;


// if "theta" is measured COUNTER-CLOCKWISE from the zero reference:

temp    =  forward*cos(theta) - right*sin(theta);
right   =  forward*sin(theta) + right*cos(theta);
forward =  temp;


// Now apply the inverse kinematic tranformation
// to convert your vehicle motion command
// to 4 wheel speed commands:

front_left  = forward + clockwise + right;
front_right = forward - clockwise - right;
rear_left   = forward + clockwise - right;
rear_right  = forward - clockwise + right;


// Finally, normalize the wheel speed commands
// so that no wheel speed command exceeds magnitude of 1:

max = abs(front_left);
if (abs(front_right)>max) max = abs(front_right);
if (abs(rear_left)>max)   max = abs(rear_left);
if (abs(rear_right)>max)  max = abs(rear_right);

if (max>1)
  {front_left/=max; front_right/=max; rear_left/=max; rear_right/=max;}


// You're done.  Send these four wheel commands to their respective wheels
```