Electr Eng (2017) 99:1161–1168
DOI 10.1007/s00202-017-0614-3

ORIGINAL PAPER

Control of the mobile robot by hand movement measured by inertial sensors

Dušan Nemec[1] · **Aleš Janota**[1] · **Michal Gregor**[1] · **Marián Hruboš**[1] · **Rastislav Pirník**[1]

Received: 16 December 2016 / Accepted: 17 July 2017 / Published online: 25 July 2017

Abstract This paper discusses methods which allow control of a mobile wheeled robot by gestures issued by human operator using movement of his arm. The gestures are measured by low-cost inertial sensors mounted on the operator's wrist and processed by algorithm proposed in this paper. Such approach allows user to control the mobile platform in outdoor environment without need for any static equipment (e.g., panel with joystick). Methods have been evaluated on E-puck mobile robot.

Keywords Remote control · Inertial measurement unit · Operator interface

1 Introduction

Recent development in the industry shows increasing deployment of mobile robotics. Autonomous ground vehicles (AGVs) are widely used in automated stores for transportation of goods and parts needed in manufacturing process. In order to achieve additional flexibility of the mobile system it is often convenient for human operators to compensate position of the vehicle according to the current situation. Similar problem occurs when mobile robot is used outdoors, when there is no pre-build HMI interface (like operator panels with joystick), and the operator cannot be in physical contact with the working robot. This requires ergonomic and easy-to-use control interface which does not require any fixed appliance. Commonly used joysticks have limitations: joystick is either mounted on the fixed platform (e.g., operator's panel) or the human operator has to hold the platform of the joystick. The first approach lacks any kind of mobility which decreases worker's efficiency. The second approach requires both hands to safely operate the joystick. Commercially available single-hand joysticks are held by fingers and operated by a thumb. These devices are meant to be used for gaming and require fine motor skills. Our method proposes usage of small low-cost inertial measurement unit (IMU) which can be mounted on the operator's wrist like a wristwatch. Such approach should eliminate the need for fine motor skills (operator does need to use fingers to operate the interface), which can be especially useful in cold weather when using fingers is not very ergonomic, or for partially disabled people who cannot move fingers.

2 Measuring attitude and heading by gyroscope, accelerometer and magnetometer

Inertial sensors (gyroscope and accelerometer) measure angular velocity and acceleration with respect to the inertial frame of reference. By nonlinear integrative algorithm it is theoretically possible to compute attitude (angle of rotation in the ground plane) and heading (rotation around the vertical axis) only from gyroscope readings. These algorithms are properly described in [1,2].

✉ Marián Hruboš
marian.hrubos@fel.uniza.sk

Dušan Nemec
dusan.nemec@fel.uniza.sk

Aleš Janota
ales.janota@fel.uniza.sk

Michal Gregor
michal.gregor@fel.uniza.sk

Rastislav Pirník
rastislav.pirnik@fel.uniza.sk

[1] Department of Control and Information Systems, Faculty of Electrical Engineering, University of Žilina, Žilina, Slovak Republic

Our current research shows low-cost sensors do not offer enough accurate readings. Real IMU systems usually utilize accelerometer as a secondary sensor because in steady state the accelerometer measures also gravity acceleration (which points in vertical direction) [3,4]. Hence the accelerometer provides enough information for attitude estimation in steady state:

$$\alpha_{\text{acc}} = \text{atan2}(-a_y, -a_z), \tag{1}$$

$$\beta_{\text{acc}} = \text{atan2}\left(a_x, \sqrt{a_y^2 + a_z^2}\right), \tag{2}$$

where α_{acc} is the roll angle (rotation around axis x) and β_{acc} is the pitch angle (rotation around axis y) in ZYX rotational convention. Acceleration components a_x, a_y, a_z are measured by accelerometer in local frame of reference (bound with the wrist). The Cartesian axes are considered to be oriented according to the North-West-Up (NWU) convention (x—forward, y—left, z—up) [5].

In order to compensate errors of the heading, we need to use magnetometer which measures magnetic field of the Earth. Since the magnetometer also rotates with the wrist in 3D space, it is necessary to transform measured magnetic field's induction vector $\boldsymbol{B}$ from rotated (local) frame of reference into horizontal plane by following formula:

$$\boldsymbol{B}_{\text{rot}} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \cdot \boldsymbol{B}. \tag{3}$$

The heading γ estimated by the magnetometer is then:

$$\gamma_{\text{mag}} = \text{atan2}(-B_{y(\text{rot})}, B_{x(\text{rot})}). \tag{4}$$

Accelerometer and magnetometer are only supplementary sensors in AHRS (Attitude and Heading Reference Systems), because the attitude angles obtained by accelerometer are strongly affected by vibrations (see Fig. 1) and error of the heading obtained by magnetometer depends on the precision of attitude (see (3)) and bias of the magnetometer. Purpose of accelerometer and magnetometer is compensation of gyroscope's bias (which is causing the drift of the angles estimated only from gyroscope) and zero reference of the attitude and heading.

3 Using IMU as a human–machine interface

Main idea of using AHRS as a human–machine interface for remote control of a mobile platform is to issue control commands according by simple gestures or moves of the operator's arm. The inertial measurement unit (IMU) can be

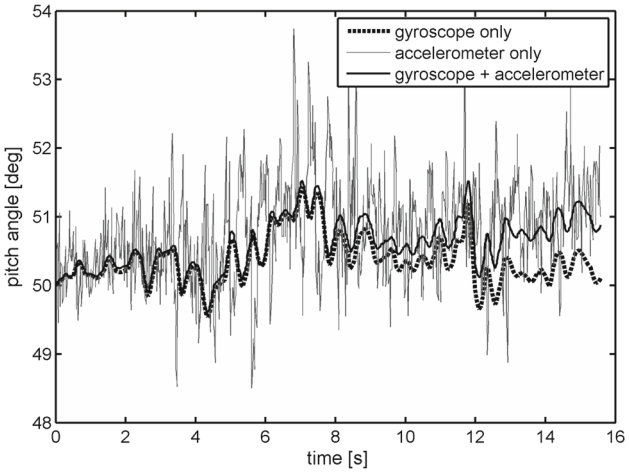

Fig. 1 Comparison between pitch angle obtained from gyroscope readings, accelerometer readings and results from sensor fusion combining accelerometer and gyroscope

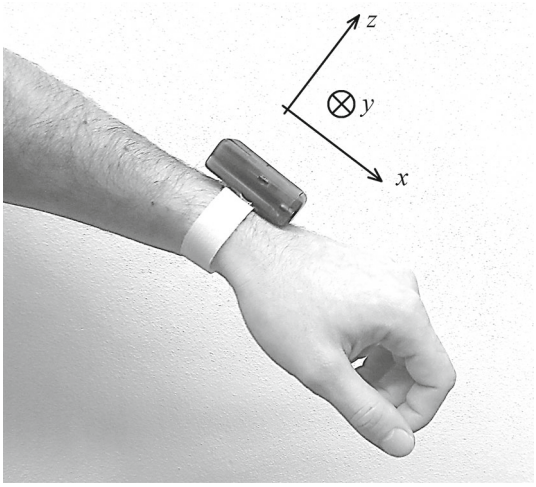

Fig. 2 Inertial measurement unit x-IMU mounted on the operator's wrist

mounted on the wrist like wristwatch (see Fig. 2). For convenience the x-axis is oriented approx. along the axis of the forearm. For navigation in the outdoor environment the intuitive gesture for human is to point desired direction by his hand in horizontal plane (see Fig. 3). The desired speed is increasing with the elevation of the operator's arm. The desired direction is determined by heading γ of the operator's arm.

Due to the human anatomy it is not ergonomic to point directly down ($\beta = 90°$); therefore, the elevation deadband β_0 has to be used. The desired speed is then:

$$v_{\text{SP}} = \begin{cases} 0 & \text{if } \beta > \beta_0, \\ v_{\max} & \text{if } \beta < 0, \\ v_{\max}\,(1 - \beta/\beta_0) & \text{otherwise}, \end{cases} \tag{5}$$

where $v_{\max}$ is the maximal desired speed. The desired velocity vector in the horizontal 2D plane is:

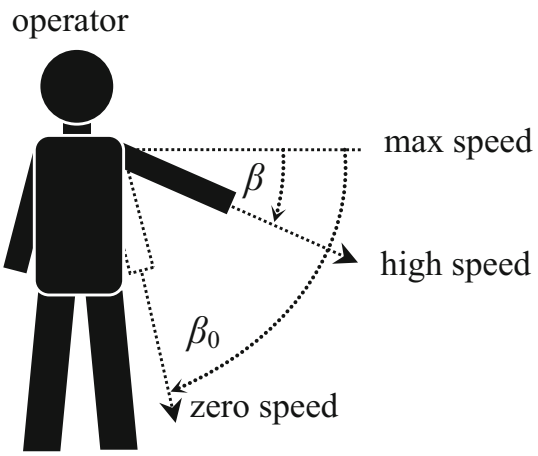

Fig. 3 Operator's gesture controlling speed of the robot

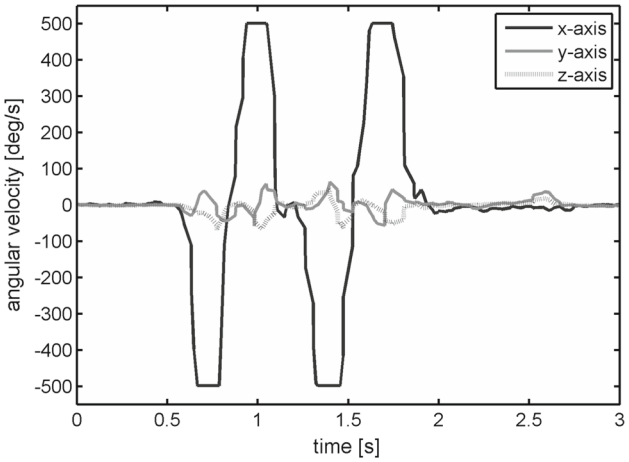

Fig. 4 Angular velocity of wrist during start/stop command

$$\boldsymbol{v}_{\mathrm{SP}} = v_{\mathrm{SP}} \cdot [\cos\gamma,\ \sin\gamma]. \tag{6}$$

3.1 Enabling/disabling the control

In order to avoid unintended commands issued when the operator rises his arm, the system should issue the commands only when operator activates it. There are two ways to activate/deactivate the system:

- mechanical button, which is pressed by operator in order to keep control system active,
- gesture recognition.

Using a button to activate the system is less ergonomic, but the button also provides safety function (once released, the robot should stop immediately). On the other hand, it is possible to choose start-up gesture which is easy to recognize programmatically and it is not likely to occur accidentally during other operator's work. According to our research, such movement is, e.g., twisting the wrist twice (around longitudinal axis of the forearm). Such movement can be characterized by large peaks of x-axis component of the measured angular velocity (see Fig. 4). Gesture can be used as a toggle switch for both enabling and disabling the system.

Pattern shown in Fig. 4 can be recognized by four peaks (angular velocity in figure saturates to full scale of the gyroscope ±500 deg/s) with alternating polarity. The polarity of the first peak depends on user and the hand on which the IMU is mounted on (left or right). The detection function *detect_enable_cmd* can be expressed by following pseudocode. The thresholds *TH_ON* and *TH_OFF* are set according to the Fig. 4. Function *now*() is supposed to return system clock in seconds. Parameters *wx*, *wy* and *wz* are components of the measured angular velocity.

```
function detect_enable_cmd(wx, wy, wz) returns boolean
    static state = 0;
    static t0 = 0;

    const TH_ON = 350;   % degrees per second
    const TH_OFF = 150;  % degrees per second
    const TIMEOUT = 2.0; % seconds

    % reset if y and z axis are not still
    if abs(wy) > TH_OFF || abs(wz) < TH_OFF
        state = 0;
    end

    switch state
        case 0
            % waiting for first peak
            if abs(wx) > TH_ON
                state = 1;
                polarity = sign(wx);
                t0 = now();
            end

        case 1
            % waiting for the peak with opposite polarity
            if abs(wx) > TH_ON && sign(wx) == -polarity
                state = 2;
            end

        case 2
            % waiting for the peak with the same polarity
            if abs(wx) > TH_ON && sign(wx) == polarity
                state = 3;
            end

        case 3
            % waiting for the peak with opposite polarity
            if abs(wx) > TH_ON && sign(wx) == -polarity
                state = 4;
            end

        case 4
            % waiting for the ending still phase
            if abs(wx) < TH_OFF
                state = 0;
                return true;
            end
    end

    % check timeout
    if (now() - t0) > TIMEOUT
        state = 0;
    end

    % in all other cases return false
    return false;
end
```

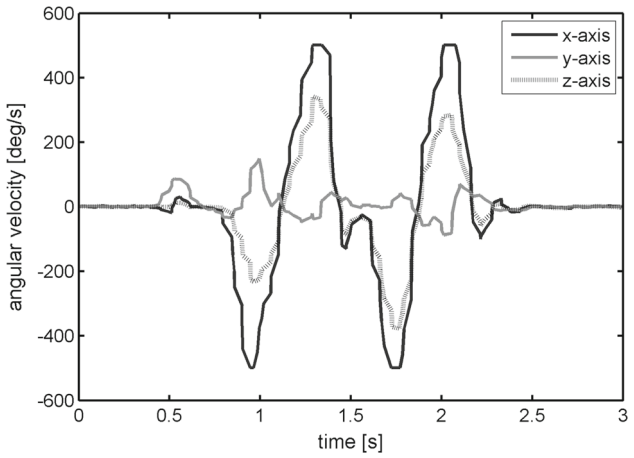

Fig. 5 Angular velocity of the enabling gesture measured by misaligned gyroscope sensor

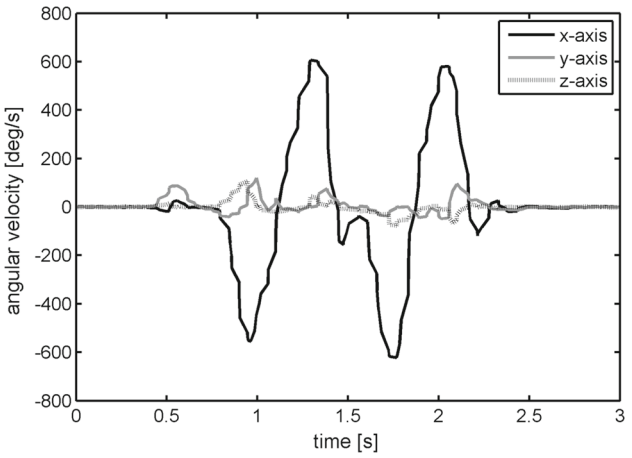

Fig. 6 Angular velocity of the enabling gesture shown in Fig. 5 after calibration

3.2 Misalignment and calibration of IMU

If the IMU is not aligned perfectly with the operator's wrist, the measured angular velocity around local y and z axes is higher than threshold (see Fig. 5). The alignment matrix $\boldsymbol{A}$ is basically a rotation matrix [6] which converts the measured angular velocity $\boldsymbol{\omega}$:

$$\boldsymbol{\omega}_{\text{out}} = \boldsymbol{A} \cdot \boldsymbol{\omega}. \tag{7}$$

In order to keep the alignment matrix normalized with unite determinant, the calibration algorithm should not estimate all its elements $A_{i,j}$. Better approach is to express it as a function of completely independent Euler angles. Since the misalignment rotation around axis x is irrelevant, the alignment matrix can be constructed from rotations around axes y and z by angles a and b:

$$\begin{aligned}\boldsymbol{A} &= \begin{bmatrix} \cos b & 0 & \sin b \\ 0 & 1 & 0 \\ -\sin b & 0 & \cos b \end{bmatrix} \cdot \begin{bmatrix} \cos c & -\sin c & 0 \\ \sin c & \cos c & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos b \cdot \cos c & -\cos b \cdot \sin c & \sin b \\ \sin c & \cos c & 0 \\ -\sin b \cdot \cos c & \sin b \cdot \sin c & \cos b \end{bmatrix}. \end{aligned} \tag{8}$$

Due to the trigonometric functions, the matrix is nonlinear and it is not possible to use linear deterministic least-square calibration methods. On the other hand, we can use stochastic methods based on gradient descent algorithm. One of the best modification is ADAM algorithm described in [7]. The learning goal is to minimize angular velocity of the measured control pattern in yz plane expressed by loss function E:

$$E = \frac{1}{2} \sum_k \left(\omega^2_{y(\text{out}),k} + \omega^2_{z(\text{out}),k} \right). \tag{9}$$

where k is the index of the sample in whole measured pattern. Combining (7)–(9) we obtain:

$$\begin{aligned} E = {} & \frac{1}{2} \sum_k \left(\omega_{x,k} \sin c + \omega_{y,k} \cos c \right)^2 \\ & + \frac{1}{2} \sum_k \left(-\omega_{x,k} \sin b \cos c + \omega_{y,k} \sin b \sin c \right. \\ & \left. + \omega_{z,k} \cos b \right)^2 . \end{aligned} \tag{10}$$

Derivatives of the loss function by parameters b, c are:

$$\begin{aligned} \frac{\partial E}{\partial b} = {} & \sum_k \left(\omega_{x,k} \cos b \cos c - \omega_{y,k} \cos b \sin c + \omega_{z,k} \sin b \right) \cdot \\ & \cdot \left(\omega_{x,k} \sin b \cos c - \omega_{y,k} \sin b \sin c - \omega_{z,k} \cos b \right), \end{aligned} \tag{11}$$

$$\begin{aligned} \frac{\partial E}{\partial c} = {} & \sum_k \left(\omega_{x,k} \sin c + \omega_{y,k} \cos c \right) \cdot \left(\omega_{x,k} \cos^2 b \cos c \right. \\ & \left. - \omega_{y,k} \cos^2 b \sin c + \omega_{z,k} \cos b \sin b \right). \end{aligned} \tag{12}$$

These derivatives are the input for ADAM algorithm. Resultant angular velocity after calibration is shown in Fig. 6. Obtained parameters for raw gyroscope data shown in Fig. 5 after 1500 iterations are $b = 0.525\,\text{rad}$ and $c = 0.076\,\text{rad}$. The calibration parameters are different for each user.

4 Control algorithm for non-holonomic mobile platform

Mobile robot is holonomic if it is capable to move in any direction and rotate around its center. Such behavior requires advanced construction of the wheels (so called Swedish

wheels) or fully steered wheels, which might be problematic in certain applications. If we want to avoid sliding drive (which is causing excessive wear of the tires), Ackerman or differential drive can be used. The main constraint of the Ackerman chassis (car-like) is its minimal turning radius $R_{\min}$. Such mobile platform has two control inputs: throttle T and steering S. Both control inputs are dimensionless numbers in the range $<-1, 1>$. The simplified mathematic model of movement of such chassis is:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = v \cdot \cos\psi, \tag{13}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = v \cdot \sin\psi, \tag{14}$$

$$\frac{\mathrm{d}\psi}{\mathrm{d}t} = \frac{v}{R_{\min}} S, \tag{15}$$

$$\frac{\mathrm{d}v}{\mathrm{d}t} = \frac{v_{\max} T}{\tau_{\mathrm{drive}}} - \frac{v}{\tau_{\mathrm{drive}}}, \tag{16}$$

where t is time, x and y are global coordinates of the vehicle's origin in the 2D plane (origin of the vehicle is the centrum of its non-steered axle), ψ is the heading of the vehicle, v is the forward speed of the vehicle's origin, $v_{\max}$ is maximal considered speed, τ_{drive} is the time constant of the drive expressing its inertia.

The goal of the control algorithm is to transform user inputs (speed setpoint v_{SP} and heading setpoint γ) to the control commands (steering S and throttle T) for the mobile platform according to the measurements of the actual heading and velocity. The heading of the vehicle can be measured by another IMU, and speed of the vehicle can be easily obtained from odometers.

4.1 Heading control with variable gain

According to (15) and (16) the control of the heading depends on the control of the speed v. If we assume that the changes of the speed are much slower than changes of the heading, the speed in Eq. (15) can be considered constant. Then the discrete transfer function in z-domain of the steering control is:

$$F_S(z) = \frac{\Psi(z)}{S(z)} = \frac{v}{R_{\min}} \cdot \frac{T_{\mathrm{S}}}{1 - z^{-1}}, \tag{17}$$

where T_{S} is the sampling period of the system. Simplest control algorithm is proportional closed-loop regulator with gain K_{P} (see Fig. 7).

Difference block in Fig. 7 subtracts two angles in radians, and it has to remove period in order to keep angular error δ in the range $<-\pi, \pi>$:

$$\delta = \gamma - \psi - 2\pi \cdot \left\| \frac{\gamma - \psi}{2\pi} \right\|, \tag{18}$$

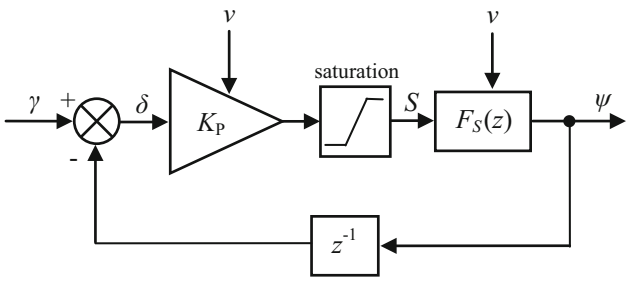

Fig. 7 Steering control scheme

where $\|x\|$ denotes the nearest integer value of x.
The transfer function of the whole closed-loop control system is (considering unit delay in the feedback):

$$\begin{aligned} F(s) &= \frac{K_{\mathrm{P}} F_S(z)}{1 + K_{\mathrm{P}} F_S(z) z^{-1}} \\ &= \frac{K_{\mathrm{P}} v T_{\mathrm{S}}}{R_{\min}\left(1 - z^{-1}\right) + K_{\mathrm{P}} v T_{\mathrm{S}} z^{-1}} \\ &= \frac{K_{\mathrm{P}} v T_{\mathrm{S}} z}{R_{\min} z + K_{\mathrm{P}} v T_{\mathrm{S}} - R_{\min}}. \end{aligned} \tag{19}$$

The system is aperiodic (without overshoot) when roots of the denominator are real and from the interval $<0,1>$. The root is:

$$z_0 = 1 - \frac{K_{\mathrm{P}} v T_{\mathrm{S}}}{R_{\min}}. \tag{20}$$

The gain of the controller has to meet following conditions:

$$K_{\mathrm{P}} v \geq 0 \quad \wedge \quad K_{\mathrm{P}} \leq \frac{R_{\min}}{v T_{\mathrm{S}}}. \tag{21}$$

Higher gain provides faster convergence so it is better to choose it at the border of the aperiodic criteria as a function of the speed v:

$$K_{\mathrm{P}}(v) \sim \frac{R_{\min}}{v T_{\mathrm{S}}}. \tag{22}$$

At zero speed the gain converges to infinity. Since the angular error δ and the controller output S are limited, the gain also should be limited to some user-chosen value $K_{\mathrm{P(max)}}$:

$$K_{\mathrm{P}}(v) = \min\left(\frac{R_{\min}}{v T_{\mathrm{S}}}, K_{\mathrm{P(max)}}\right). \tag{23}$$

In order to meet criteria at full range of speed the controller gain can be set to fixed value (considering maximal positive speed $v_{\max}$):

$$K_{\mathrm{P(min)}} = \frac{R_{\min}}{v_{\max} T_{\mathrm{S}}}. \tag{24}$$

4.2 Speed control

The approximated discrete transfer function of the throttle control derived from (16) is following:

$$F_T(z) = \frac{V(z)}{T(z)} = \frac{v_{\max} T_S}{T_S + \tau_{\text{drive}} \left(1 - z^{-1}\right)}. \tag{25}$$

For such system we might use PI-like closed-loop discrete controller [9–11]. However, if we consider the power consumption of the propulsion system it is not desirable to minimize the convergence time of the control. There are many approaches for optimization of the control [12–14]. The control system highly depends on the construction of the propulsion system so it cannot be generalized (Eq. (16) is very schematic).

4.3 High-level control considering on-board proximity sensors

There are four approaches how the non-holonomic robot can align with desired heading (see Fig. 8). Note that ways III. and IV. are reverse.

The mentioned ways differ by their costs. The cost of the selected way depends on:

- previously selected way (changing the direction takes time),
- angular error (higher angular error means longer trajectory),
- presence of the obstacle measured by proximity sensors.

The cost from angular error can be expressed by following:

$$E_{\text{I}} = d_{\text{fwd}} \cdot \left[C_{\text{turn}} |\delta| + E_{\text{rev}} \text{neg}(v)\right], \tag{26}$$

$$E_{\text{II}} = d_{\text{bck}} \cdot \left[C_{\text{turn}} |\delta| + E_{\text{rev}} \text{pos}(v) + E_{\text{rev}}\right], \tag{27}$$

$$E_{\text{III}} = d_{\text{fwd}} \cdot \left[C_{\text{turn}} (\pi - |\delta|) + E_{\text{rev}} \text{neg}(v) + E_{\text{rev}}\right], \tag{28}$$

$$E_{\text{IV}} = d_{\text{bck}} \cdot \left[C_{\text{turn}} (\pi - |\delta|) + E_{\text{rev}} \text{pos}(v)\right], \tag{29}$$

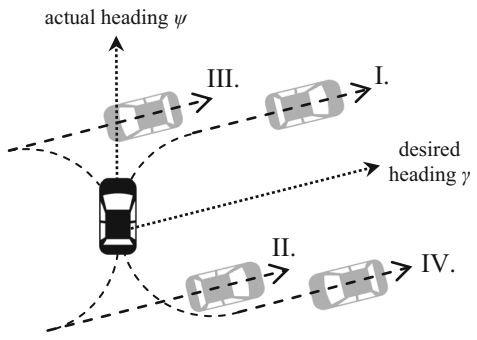

Fig. 8 Four ways to achieve the same heading

where C_{turn} is the cost of the turning per radian, E_{rev} is the cost of the reversing; function $\text{pos}(v)$ returns 1 when $v > 0$ otherwise returns 0; function $\text{neg}(v)$ returns 1 when $v < 0$, otherwise returns 0. Variables d_{fwd} and d_{bck} reflect danger of the collision if the robot will move forward or backward, respectively. If there is no obstacle $d = 1$, if the obstacle is near $d \to \infty$.

The control algorithm is written by following pseudocode:

```
enable = false;
connect IMU;
connect robot;

while true
    read IMU.[wx wy wz beta gamma];
    read robot.[v psi proximity];

    % check enable / disable commands
      if detect_enable_cmd(wx, wy, wz)
         enable = !enable;
      end

      if enable
         compute v_SP;
         compute delta;
         compute E_I, E_II, E_III, E_IV;
         E_min = min(E_I, E_II, E_III, E_IV);

         if E_I == E_min
              T = T_controller(v, +v_SP);
              S = S_controller(psi, gamma);

         elseif E_II == E_min
              T = T_controller(v, -v_SP);
              S = -S_controller(psi, gamma+pi);

         elseif E_III == E_min
              T = T_controller(v, +v_SP);
              S = S_controller(psi, gamma+pi);

         elseif E_IV == E_min
              T = T_controller(v, -v_SP);
              S = -S_controller(psi, gamma);
         end

      else
         T = 0;
         S = 0;
      end

      write robot.[T S];
end
```

The functions *T_controller* and *S_controller* implement control algorithms mentioned in Sects. 4.1 and 4.2.

5 Hardware used for evaluation

In our experiments we have used commercially available product *x-IMU* manufactured by *x-IO Technologies Ltd*. This inertial measurement unit integrates gyroscope, accelerometer, magnetometer and data-processing CPU. All processed

data are sent wirelessly by Bluetooth interface to the PC (acts like virtual COM port).

As an evaluation mobile platform we have used E-puck mobile robot [8], which incorporates necessary proximity sensors and encoders. Since it is differentially driven 2-wheeled robot (see Fig. 9), it was programmatically constrained to behave like Ackerman chassis (the nonzero minimal turning radius was introduced).

6 Experimental evaluation

The evaluation of any human–machine interface is usually a complex problem. In order to evaluate our proposed approach, we have chosen to compare the time which is needed by human user to control the robot along pre-defined trajectory (see Fig. 10). The length of the optimal trajectory via all waypoints is approx. 4.4 m, the maximal speed of the robot is 0.13 m s^{-1}. The best possible time is 33.8 s (Table 1).

Experiment participants were instructed to drive (control) the robot manually via all waypoints in given order using different HMI interfaces. Our interface using IMU is compared with a standard joystick (gaming device) and a keyboard. Each interface was evaluated six times in order to compare the training capabilities of the participants. The average results of 43 participants are shown in Table 1.

Fig. 9 E-puck mobile robot

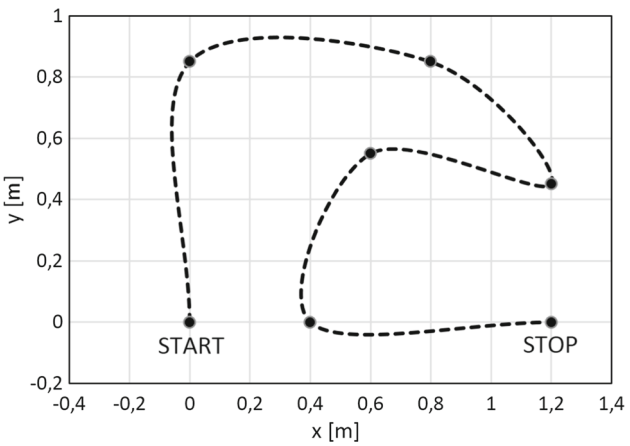

Fig. 10 Experimental trajectory

Table 1 Average running time in seconds of all participants using different interfaces

Method/attempt:	1	2	3	4	5	6	All
IMU (proposed)	43.6	40.5	39.4	38.3	37.7	35.9	39.2
Joystick	45.2	38.2	39.5	40.0	41.0	40.3	40.7
Keyboard	56.6	58.2	52.2	52.3	51.4	51.3	53.7
All	48.5	45.6	43.7	43.5	43.4	42.5	

Table 2 Standard deviation of the average time in seconds for all participants using different interfaces

Method/attempt:	1	2	3	4	5	6	All
IMU (proposed)	2.4	5.5	3.1	1.9	2.4	1.0	3.7
Joystick	8.7	5.4	4.8	5.5	9.7	7.6	6.8
Keyboard	9.0	7.8	4.7	4.4	4.0	4.7	6.2
All	9.1	10.6	7.4	7.6	8.3	8.3	

As can be seen, our interface achieved the best results (shortest time). If we compare the standard deviation of the average time (Table 2), the deviation of the standard joystick is higher than IMU since the heading reference system inside IMU system allows independent 3rd person view and the operator is capable to control the movement just by movement of his/her body. On the other side, the joystick and keyboard controlled the robot in 1st person view mode, which were a little more difficult for some participants.

7 Conclusion

This article analyzed possibility of using inertial measurement unit as a human-to-machine interface for mobile robot control. Such interface does not require both hands for safe operation and increases mobility of the human operator. Because it incorporates heading reference system, it takes into account also current orientation of the operator. Our practical experience with the experimental platform has shown that such interface does not require long training and is quite intuitive. Interface can be used in industrial solutions or for outdoor control of the mobile robot; its deployment in the area of health care and mobility of disabled peoples will be a subject for our next studies.

Acknowledgements This work was supported by Slovak Research and Development Agency under the Project No. APVV-15-0441.

References

1. Janota A, Šimák V, Nemec D, Hrbček J (2015) Improving precision and speed of Euler angles computing from low cost sensor data. Sensors 15(3): s.7016–7039. doi:10.3390/s150307016. ISSN 1424-8220
2. Hulsing R (1998) MEMS inertial rate and acceleration sensor. IEEE Aerosp Electr Syst Mag 13(11): 17–23. doi:10.1109/62.730613. ISSN: 0018-9251
3. Benser ET (2015) Trends in inertial sensors and applications. In: IEEE international symposium on inertial inertial sensors and systems (ISISS), pp 1–4, 23–26 March 2015. doi:10.1109/ISISS.2015.7102358
4. Pinney C, Hawes MA, Blackburn J (1994) A cost-effective inertial motion sensor for short-duration autonomous navigation. In: Position location and navigation symposium, 1994, IEEE, pp 591–597, 11–15 Apr 1994. doi:10.1109/PLANS.1994.303402
5. Sampaio S, Massatoshi Furukawa C, Maruyama N (2013) Sensor fusion with low-grade inertial sensors and odometer to estimate geodetic coordinates in environments without GPS signal. IEEE Latin Am Trans (Revista IEEE America Latina), 11(4): 1015–1021. doi:10.1109/TLA.2013.6601744. ISSN: 1548-0992
6. Qasem H, Gorgis O, Reindl L (2008) Design and calibration of an inertial sensor system for precise vehicle navigation. In: 5th Workshop on positioning, navigation and communication, 2008. WPNC 2008. pp 229–231, 27–27 March 2008. doi:10.1109/WPNC.2008.4510379. ISBN 978-1-4244-1798-8
7. Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: International conference on learning representations. arXiv:1412.6980
8. Mondada F et al (2009) The e-puck, a robot designed for education in engineering. In: Proceedings of the 9th conference on autonomous robot systems and competitions, vol 1(1), pp 59–65
9. Arrieta O, Vilanova R, Rojas JD et al (2016) Improved PID controller tuning rules for performance degradation/robustness increase trade-off. Electr Eng. doi:10.1007/s00202-016-0361-x
10. Arab Markadeh G, Soltani J (2006) Robust direct torque and flux control of adjustable speed sensorless induction machine drive based on space vector modulation using a PI predictive controller. Electr Eng. doi:10.1007/s00202-005-0310-6
11. Polajžer B, Ritonja J, Štumberger G et al (2005) Decentralized PI/PD position control for active magnetic bearings. Electr Eng. doi:10.1007/s00202-005-0315-1
12. Can H, Akin E (2002) Neural network-based stator voltage compensator for low-frequency operation of a vector-controlled induction motor drive. Electr Eng. doi:10.1007/s00202-002-0132-8
13. Shi ZY, Zhong YS, Xu WL (2004) Decentralized robust tracking control for uncertain robots. Electr Eng. doi:10.1007/s00202-004-0232-8
14. Huang YJ, Kuo TC (2004) Robust output tracking control for nonlinear time-varying robotic manipulators. Electr Eng. doi:10.1007/s00202-003-0218-y