Design and Control of Self-Stabilizing Angular Robotics Anywalker

Igor Ryadchikov, Semyon Sechenev, Sergey Sinitsa, Alexander Svidlov, Pavel Volkodav, Anton Feshin, Anas Alotaki, Aleksey Bolshakov, Michail Drobotenko
Kuban State University,
Krasnodar, Russia

Evgeny Nikulchev
Moscow Technological Institute
Moscow, Russia

***Abstract*—Walking robots are designed to overcome obstacles when moving. The walking robot AnyWallker is developed, in the design of which the task of self-stabilization of the center of the mass is solved; a special type of chassis is developed, providing movement on high cross-country capability. The paper presents the results of designing and controlling the robot, the architecture of the software complex provides management and mastification of the hardware platform. AnyWalker is actually a chassis which can be used to build robots for many different purposes, such as surveying complex environment, industrial operations, and work in hazardous environment.**

Keywords—Walking robots; self-stabilization platform; stability of dynamic systems; chassis of robotic complexes

I. Introduction

A significant number of moving transport robotic complexes are wheeled or caterpillar. However, such robots become practically useless in rough terrain, in rooms with stairs and a large number of obstacles [1]. This is especially important in hazardous environments, in mines or where it is necessary to preserve the landscape. Many studies in recent years are aimed at implementing and studying the quality characteristics of walking robot designs [2].

The restriction imposed on overcoming obstacles by wheeled robots directly depends on the diameter of the wheel. While walking robots can overcome obstacles and are limited by the length of the leg. Another important advantage of walking robots is that only local horizontal supports should be present in the terrain. This allows overcoming very steep angles while maintaining the stability of the body [3]. Walking robots were used to investigate remote locations and hostile environments, such as the seabed, space, nuclear power plants, and in rescue operations [4]. In addition, vehicles with a walking principle can be used, for example, for collecting materials, for transporting goods, as service robots, for moving to hard-to-reach areas of production and main pipelines. According to the leading US universities and companies, expressed in the report A Roadmap for US Robotics 2016: “To extend the automation of the logistics chain into the world, robots must have mobility that matches human mobility – robots must negotiate stairs, elevators, doorways, curbs, broken concrete, cluttered environments, and go where people go. This type of advanced mobility is becoming realistic for robotic systems, legged and otherwise - and with such a solution, logistics will become fast, 24/7, on-demand, inexpensive, predictable, and well-tracked.”

Despite a wide range of applications, many tasks remain unresolved, which makes it difficult to widely use walking robots. The drawbacks include: high complexity of control and stabilization, cost, low energy efficiency and relatively low speed. Modern research is concerned with these issues [5].

In the basis of the movement of walking systems, the principles of movement by man and animals are laid: step, jogging, jumping [6]. By design, we can distinguish a class of humanoid robots, the advantage of which is the reduction of energy costs due to the use of natural oscillations, but a poorly solved problem here is the stabilization of the center of mass due to the complex geometry of the object [7]. Another big class are six-legged mobile platforms with various types of chassis [4]. There are other developments: from the repetition of kangaroo jumps [8] to tripod systems [9].

The report is devoted to the design of the walking robot, the development of the control software and hardware of the mobile robot AnyWallker. In the presented design, the task was to design a robot in an easily controlled self-stabilizing platform, with a large range of patency (overcoming high and complex obstacles). The solution of these problems is ensured by the design of the hull, which allows for quick-setting stabilization, as well as the original chassis scheme aimed at overcoming obstacles.

II. Construction and Kinematic Scheme

As a body (a stabilizing center of mass), a sphere with a diameter of 0.4 m was selected (Fig. 1).

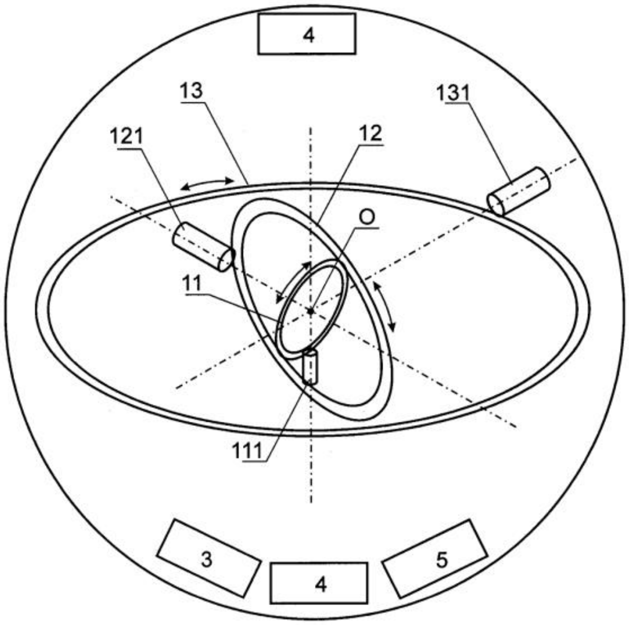

Fig. 1. Schematic arrangement of device elements in space.

Stabilization of the body position is carried out due to the force compensation of the deflecting influences. The design of this device includes two flywheels, flywheel actuators, a control unit, body position sensors, flywheel speed sensors and a power source. Both flywheels are inside the stabilized body. In this case, the mass centers of the flywheels coincide, and their axes of rotation intersect. External actions are recorded by the body position sensors and transmitted to the control unit, sending the corresponding signal to the flywheel drives and thereby driving them. The moment of inertia of the rotating flywheels compensates for the deviations caused by external influences, stabilizing the position of the body in space [10].

The stabilization device includes two flywheels 11 and 12 with flywheel actuators 111 and 121 connected to a volumetric body, a control unit 3 connected to position sensors 4, a power supply 5 and flywheel actuators 111 and 121. The rotation axes of all flywheels 11 and 12 intersect at one point O (Fig. 1). Each flywheel has a flywheel position sensor and/or an angular speed sensor connected to the control unit. Alternatively, each flywheel drive has a flywheel position sensor, and/or a flywheel angular velocity sensor connected to the control unit. The flywheel actuators 111, 121, 131 can be made in the form of electric motors, including having their own controllers and servo drives, are shown as 112, 122, 132.

A kinematic scheme of the chassis [11] has been developed, each of the robot legs comprising an upper link for supporting the volumetric body through the first hinge, and a lower link for supporting the upper link through the second hinge, and also the feet movably connected to each lower link through the third hinge. The upper link consists of two parts, the first and the second, connected to each other through a fourth hinge, the first part of the upper link of the body being connected to the body by the first hinge and the second part of the upper link through the second hinge connected to the lower link, robot standing on two legs, the fourth hinge of each robot's leg is located above the first, second and third hinges (see Fig. 2).

Fig. 2 shows the following parts of the walking robot: 1 is the body of the robot, 5 is the module for correcting the displacement of the center of mass of the robot, 6 is the surface, 8 is the axis of gravity passing through the center of mass, 9 is the obstacle, 11 is the first part of the upper leg of the robot, 12 is the second part of the upper leg of the robot, 13 is the lower leg of the robot, 14 is the foot, 21 is the first hinge, 22 is the second hinge, 23 is the third hinge, 24 is the fourth hinge.

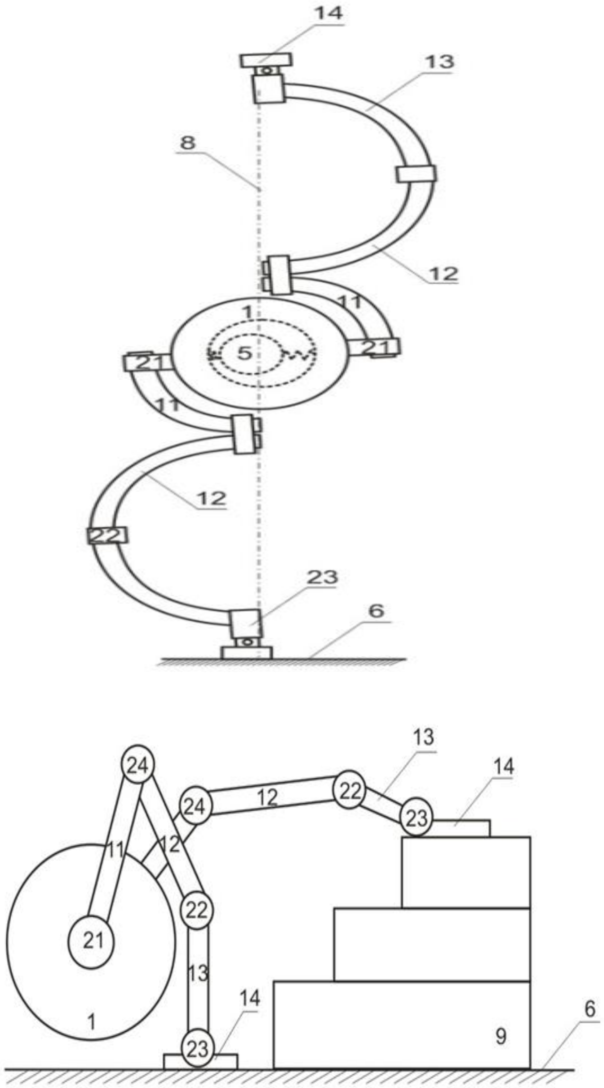

Fig. 2. Kinematic scheme.

The calculated speed of moving the platform with the dimensions of the body, fitting into the sphere of 40 cm, along the horizontal, on the average surface, in the step mode will be 5 km/h, and in the rolling mode on the body with the manipulators up to 15 km/h, it can autonomously function for 24 hours in the video broadcast mode, active control and moving through the terrain of medium cross-country, while carrying 3.8 kg payload [12].

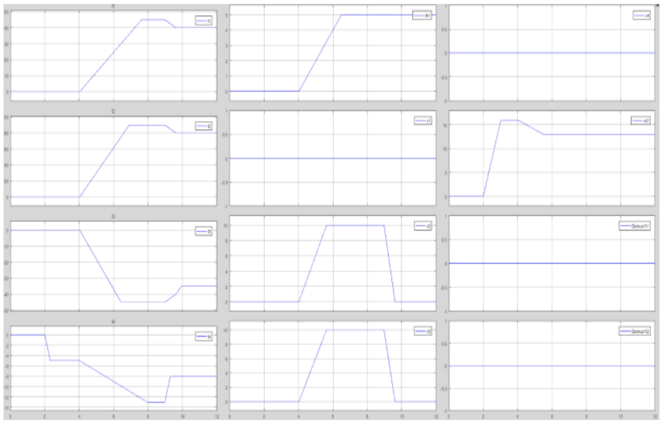

Fig. 3. Assignments to servos for a single half-step.

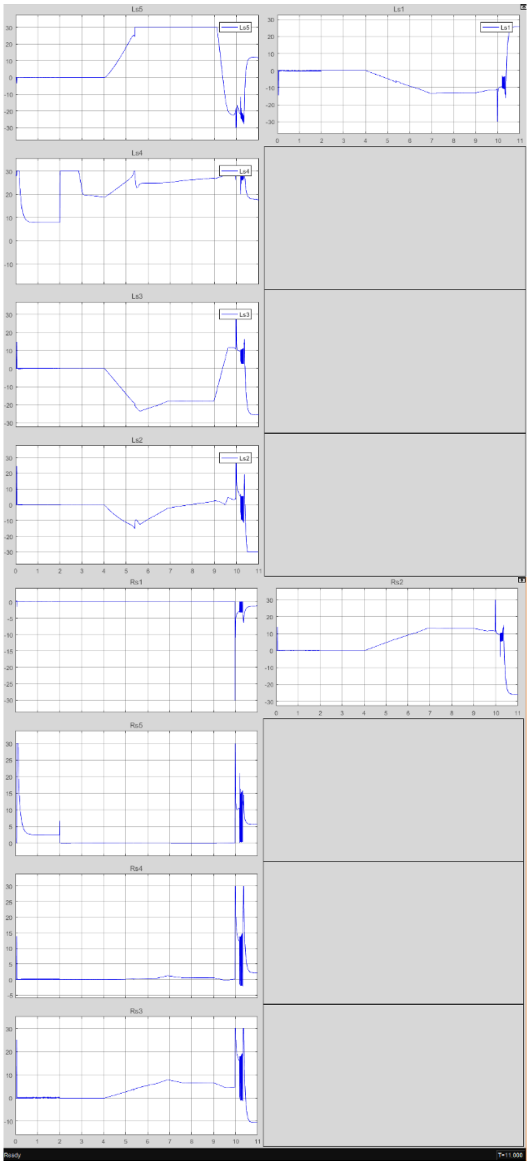

Fig. 4. The moment of force developed in the servo drives when making a half step.

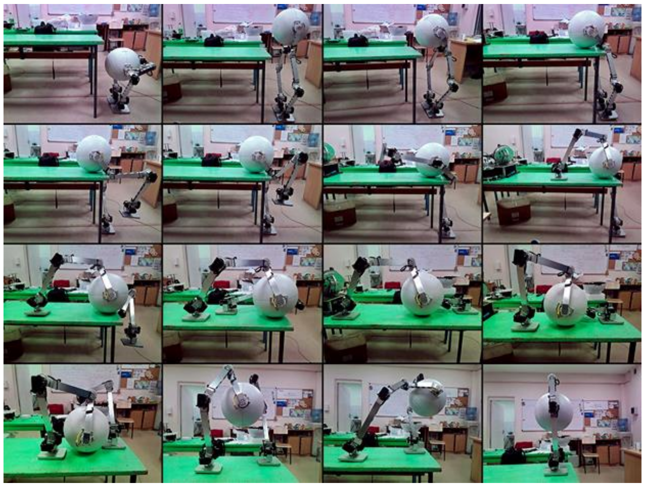

Fig. 5. Phases of overcoming the robot step, with a height equal to the height of the robot.

In Fig. 3 there are graphs of the task of the angle to the servos of the legs. The graphs with index l correspond to the left leg, with the index r to the right. The moments of force developed in the servo drives when making a half step are shown in the graphs in Fig. 4. An example of overcoming a robot step, with a height equal to the height of the robot is shown in Fig. 5.

III. Self-Stabilization of the Platform

To stabilize the walking robot, algorithms based on the following model are implemented. The ball, which is the basis of the walking robot, is regarded as an inverted pendulum with a flywheel. This was inspired by the Cubli robot presented in [17]. The pendulum has a non-holonomic connection with the support surface. This makes it possible to find a solution to the problem of stabilizing the mobile structure in dynamics, by adding internal degrees of freedom. It is proposed to assign the stabilization tasks to a system of two flywheels operating as an ordinary pendulum with a flywheel [13], [14], but in contrast to these studies, it is proposed to combine the mass centers of two flywheels at one point and obtain a more compact design (see Fig. 6).

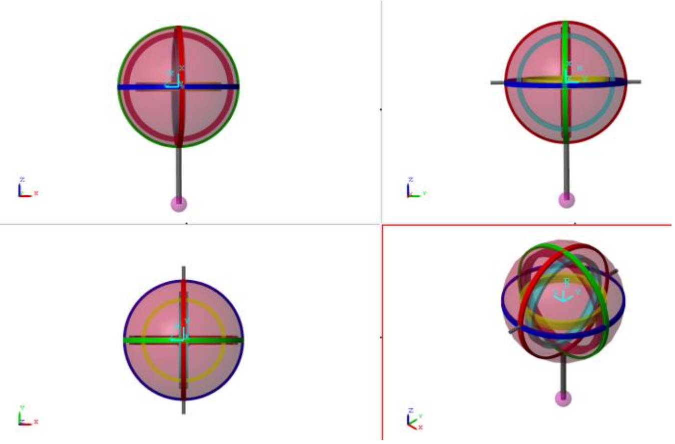

Fig. 6. Simulation model of self-stabilizing platform (inverted pendulum with flywheel).

Applying the approach described in [14], the dynamics of the system can be described as follows:

$$\begin{cases} J\chi\ddot{\beta}+\left(J_r+\chi J_M\right)\dot{\omega}=\left(mb+Ml\right)g\chi\sin\beta-\grave{u}\ \dot{\beta}, \\ \left(J_r+\chi J_M\right)\chi\ddot{\beta}+\left(J_r+\chi^2 J_M\right)\dot{\omega}=\chi L, \\ L=c_u u-c_v\dfrac{\omega}{\chi}. \end{cases} \tag{1}$$

Here the states of the system are: $\beta(t)$ is the angle between the axis of the pendulum and the direction "up" counter-clockwise; $\omega(t)$ is the angular speed of rotation of the flywheel; control: u is the voltage applied to the motor [B]; design parameters of the model: J_m is the moment of inertia of the pendulum relative to its axis of rotation [kg·m^2]; J_r is the moment of inertia of the rotor of the engine with respect to its axis of rotation [kg·m^2]; χ is the reduction ratio; J_M is the moment of inertia of the flywheel relative to its axis of rotation [kg·m^2]; m is the mass of the pendulum [kg]; *b* is the distance from the point of pendulum fixation to its center of mass [m]; *M* is the total mass of flywheel and engine [kg]; l is the length of the pendulum [m]; g is the acceleration of gravity [m·s^{-2}]; $\grave{u}$ is the coefficient of friction [N·m·s]; c_u [N·m·B^{-1}]; c_v [N·m·s]; $J=J_m+Ml^2+J_r+J_M$.

Linearized equation, taking into account the assumptions that there is no reducer $(\chi=1$ and $J_M=0)$; the intrinsic mass of the pendulum is small ($m = 0$ and $J_m=0$); the friction in the support is insignificant ($\grave{u}\ =0$), looks like:

$$\frac{d}{dt}\begin{pmatrix}\beta\\ \dot{\beta}\\ \omega\end{pmatrix}=\begin{bmatrix}0 & 1 & 0\\ \frac{g}{l} & 0 & \frac{c_v}{Ml^2}\\ -\frac{g}{l} & 0 & -\left(\frac{1}{J_r}+\frac{1}{Ml^2}\right)c_v\end{bmatrix}\begin{pmatrix}\beta\\ \dot{\beta}\\ \omega\end{pmatrix}+\begin{bmatrix}0\\ -\frac{c_u}{Ml^2}\\ \left(\frac{1}{J_r}+\frac{1}{Ml^2}\right)c_u\end{bmatrix}u.$$

or

$$\dot{x}=Ax+bu. \tag{2}$$

The characteristic polynomial of the matrix *A* has the form:

$$F\left(\lambda\right)=\begin{vmatrix}-\lambda & 1 & 0\\ \frac{g}{l} & -\lambda & \frac{c_v}{Ml^2}\\ -\frac{g}{l} & 0 & -\left(\frac{1}{J_r}+\frac{1}{Ml^2}\right)c_v-\lambda\end{vmatrix}=$$

$$=\left(-\left(\frac{1}{J_r}+\frac{1}{Ml^2}\right)c_v-\lambda\right)\left(\lambda^2-\frac{g}{l}\right)-\frac{c_v}{Ml^2}\frac{g}{l}.$$

The matrix has one positive root and two negative roots, which follows from the following facts:

$$F(0)=\tfrac{c_v g}{J_r l}>0, \qquad F\left(-\sqrt{\tfrac{g}{l}}\right)=F\left(\sqrt{\tfrac{g}{l}}\right)=-\tfrac{c_v}{Ml^2}\cdot\tfrac{g}{l}<0,$$

$F(\lambda)\to+\infty$ at $\lambda\to-\infty$. The eigenvector corresponding to the eigenvalue λ can be written as $\left(c_v,c_v\lambda,Ml(\lambda^2-g)\right)^T$. The presence of a positive eigenvalue indicates the instability of the stationary point $(0,0,0)^{\mathrm{T}}$ in the absence of control.

Making a change of variables $x=Ky$, where

$$K=\begin{pmatrix}c_v & c_v & c_v\\ c_v\lambda_1 & c_v\lambda_2 & c_v\lambda_3\\ Ml\left(\lambda_1^2-g\right) & Ml\left(\lambda_2^2-g\right) & Ml\left(\lambda_3^2-g\right)\end{pmatrix}, \tag{3}$$

we get

$$K\dot{y}=AKy+bu.$$

We multiply the equation on K^{-1}:

$$\dot{y}=K^{-1}AKy+K^{-1}bu=\mathrm{diag}(\lambda_1,\lambda_2,\lambda_3)y+K^{-1}bu,$$

Let $m=K^{-1}b$, then

$$\dot{y}=\mathrm{diag}(\lambda_1,\lambda_2,\lambda_3)y+mu, \tag{4}$$

Without loss of generality, we assume that $\lambda_1>0>\lambda_2>\lambda_3$. Let $u=u(y,t)$ be such that $|u|\le u_{max}$.

Under these conditions, the following results are proved.

The system described by (4) can not go from a state with $y_1\ge|m_1|u_{max}\lambda_1$ into a state with a smaller y_1 (it can not go from a state with $y_1\le-|m_1|u_{max}\lambda_1$ to a state with a larger y_1).

There exists a control $u=u(y,t)$ such that the solution of (4) from a state with $|m_1|u_{max}\lambda_1$ tends to the equilibrium point $(0,0,0)^{\mathrm{T}}$.

The results obtained are realized in the construction of control actions.

IV. Architecture of the Software That Controls the Robot

To control the robot, a hardware/software system with a three-level architecture was developed [15]. The subsystem of the first level is implemented on the ARM microcontroller of the STM32F4 family under the control of the real-time program based on the ST HAL library. The second-level control subsystem is implemented on a single-board computer of the Raspberry Pi family running the Robot Operating System (ROS) Kinetic Kame. The third application level is the program on MATLAB, Python and the web interface. The hardware architecture of the robot AnyWalker is shown in Fig. 7.

The main elements in this architecture are the STM32F407 microcontroller, which is a peripheral controller, and the Raspberry Pi 3 microcomputer, which plays the role of a top-level controller and provides interfaces for the interaction of control software and various clients with peripheral devices.

The peripheral controller interacts with the 9-axis inertial navigation system represented by the MPU9250 chip, using the SPI protocol. The flywheels installed in the robot are controlled by the drivers of the EPOS2 motors, communication with which is carried out via the CAN protocol. The limbs of the robot are driven by the Dynamixel MX-106 servo drives, which are connected via RS-485 protocol. Servo drives are connected in series with 6 elements per leg. Both buses are connected to two independent interfaces of the UART microcontroller.

The interaction between the upper-level controller and the peripheral controller is carried out through the UART interface, through which the peripheral control commands are transmitted, and the INS sensor readings, the flywheel speed and the temperature and load values on the servos are requested. Also, the STM32F407 controller is connected via USB for debugging and downloading updated software for the controller via the STLink v2 protocol.

The scheme also provides customers implementing various elements of logic. The clients are connected to the high-level controller via an Ethernet link or via a wireless WiFi network.

The hardware layer is the basis for organizing the software layer, the architecture of which is shown in Fig. 8.

The interaction of logic nodes present on a high-level controller and various clients with peripheral nodes is carried out through a specialized communication module with a peripheral controller. The communication module with the peripheral controller is a driver that interacts the system with a lower-level controller based on the STM32F407 chip. The driver is written in C ++ language, it accepts commands coming from the modules of automatic and manual control. The interaction with the STM32F407 is carried out via the UART via the MODBUS protocol at a speed of 921600 baud, which makes it possible to achieve a control loop frequency on the order of 150-200 Hz. Using the graph structure of ROS [16], the driver sends information about the status of the robot nodes to all network members who have subscribed to receive this data. In turn, the driver is signed to receive commands for controlling flywheel drives and servo positions. The diagram of the graph structure is shown in Fig. 9.

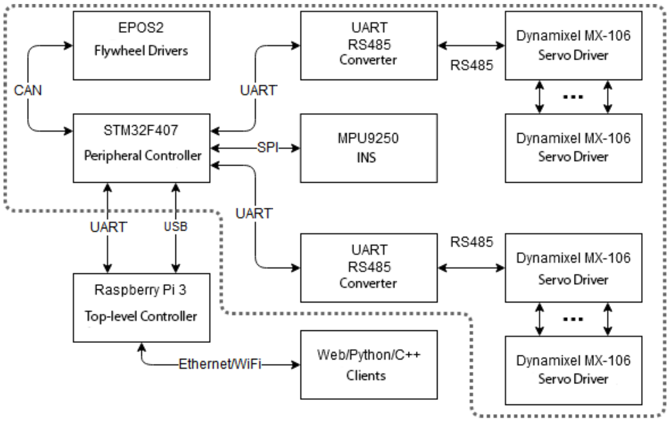

Fig. 7. Hardware architecture of the robot components.

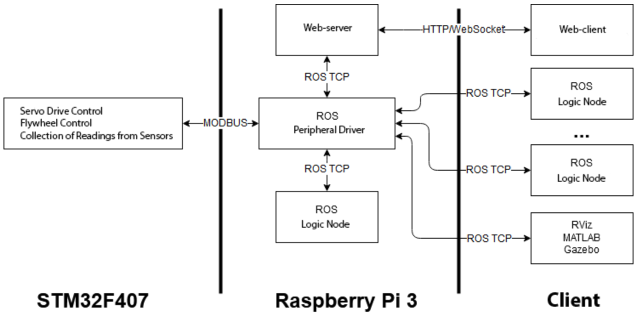

Fig. 8. The architecture of the program layer of the robot.

The driver /aw_driver continuously polls the peripheral controller, requesting the readings of the INS, servo sensors, Hall sensors from the EPOS2 drivers to monitor the speed of the flywheel and the pressure sensors on the feet. The received data are organized in a special structure, which is published in the topic /aw_driver/status/robot_status. With this data, various nodes of logic, as well as software for visualization and simulation, for example, Gazebo, RViz and MATLAB can work.

Information about the state of the robot is transmitted to a web client connected to the robot through a WiFi network or an Ethernet cable. The web client interacts with the ROS system through a two-way communication channel based on the WebSocket technology implemented by the /rosbridge_websoket node.

The network also presents the /aw_main node, which is an intermediate layer between the user and peripheral devices. This site is written in the Python language and is engaged in converting a text-based command system into a set of values intelligible to the driver, converting the readings of sensors into a user-friendly form, for example, translating the readings of servo encoders to degrees, and also calculating the speeds of the robot's nodes in automatic control mode. The node communicates with the web client via the /aw_driver/py_js (to the client) and /aw_driver/js_py (from the client) topics. The node itself refers to the driver through the system of control topics /aw_driver/control/*.

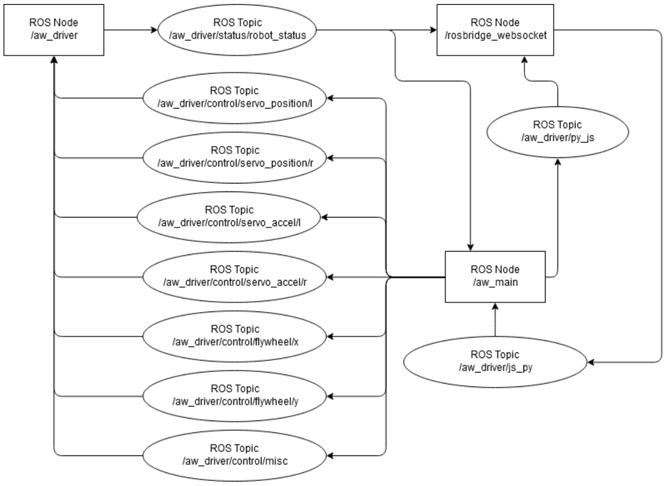
ROS Node
/aw_driver
ROS Topic
/aw_driver/status/robot_status
ROS Node
/rosbridge_websocket
ROS Topic
/aw_driver/control/servo_position/l
ROS Topic
/aw_driver/py_js
ROS Topic
/aw_driver/control/servo_position/r
ROS Topic
/aw_driver/control/servo_accel/l
ROS Topic
/aw_driver/control/servo_accel/r
ROS Node
/aw_main
ROS Topic
/aw_driver/control/flywheel/x
ROS Topic
/aw_driver/js_py
ROS Topic
/aw_driver/control/flywheel/y
ROS Topic
/aw_driver/control/misc

Fig. 9. The architecture of the ROS network presented in graphs.

V. Conclusion

The task of obtaining a stable, and energy-efficient walking robot with the ability to navigate through unknown terrain has been a big problem in the field of robotics for many years. The developed robotic platform demonstrates high energy efficiency, in comparison with other designs of walking robots [11]. The developed walking robot AnyWallker is an example of a service mobile device capable of coping with an unknown terrain, reliably and flexibly moving along the way.

Acknowledgment

The work was carried out within the framework of the state task of the Ministry of Education and Science, project No. 8.2321.2017 “Development and adaptation of control systems for compensation of dynamic deflecting effects on mobile objects in a state of dynamic equilibrium”.

References

[1] D. C. Kar, “Design of a statically stable walking robot: A review,” Journal of Robotic Systems, vol. 20, no. 11, pp. 671–686, 2003

[2] A. Goswami, “Walking Robots. Encyclopedia of Systems and Control,” pp. 1537-1548, 2015.

[3] T. Booysen and S. Marais, “The development of a remote controlled, omnidirectional six legged walker with feedback.”, In AFRICON 2013, pp. 1-6, 2013.

[4] F. Tedeschi and G. Carbone, “Design issues for hexapod walking robots,” Robotics, vol. 3, no. 2, pp. 181-206, 2014.

[5] X. Xiong, F. Wörgötter, and P. Manoonpong, “Adaptive and energy efficient walking in a hexapod robot under neuromechanical control and sensorimotor learning,” IEEE transactions on cybernetics, vol. 46, no. 11, pp 2521-2534, 2016.

[6] N. S. Szczecinski, A. J. Hunt, and R. D. Quinn, “Design process and tools for dynamic neuromechanical models and robot controllers,” Biological cybernetics, vol. 111, no. 1, pp. 105-127, 2017.

[7] J. Zhao, Q. Liu, S. Schütz and K. Berns, “Experimental verification of an approach for disturbance estimation and compensation on a simulated biped during perturbed stance,” 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 5082-5087, 2014.

[8] G. H. Liu, H. Y. Lin, H. Y. Lin, S. T. Chen, and P. C. Lin, “Design of a kangaroo robot with dynamic jogging locomotion,” 2013 IEEE/SICE International Symposium on System Integration (SII), pp. 306-311, 2013.

[9] Borràs, J., & Dollar, A. M. (2012) Static analysis of parallel robots with compliant joints for in-hand manipulation. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3086-3092.

[10] Patent RU160949U1. (2016/04/10).

[11] I. Ryadchikov, S. Sechenev, A. Svidlov, S. Sinitsa, Z. Buskandze and E. Nikulchev, “AnyWalker: all-terrain robotic chassis. In 47st International Symposium on Robotics;” Proceedings of ISR 2016, pp. 696-701, 2016.

[12] I. Ryadchikov, S. Sechenev, A. Svidlov, S. Sinitsa and E. Nikulchev, “Development of a self-stabilizing robotic chassis for industry,” MATEC Web of Conferences, vol. 99, art. 02007, 2017

[13] M. Shahbazi, R. Babuška and G. A. Lopes, “Unified modeling and control of walking and running on the spring-loaded inverted pendulum,” IEEE Transactions on Robotics, vol. 32, no. 5, pp. 1178-1195, 2016.

[14] A. M. Formalsky, Controlling the motion of unstable objects, Moscow, 2013. [In Rus]

[15] I. Riadchykov, S. Sechenev, S. Sinitsa and E. Nikulchev, “Constructive solution of the robotic chassis AnyWalker,” ITM Web of Conference, vol. 6, no. 01003, 2016.

[16] http://wiki.ros.org/ROS/Concepts#ROS_Computation_Graph_Level

[17] M. Gajamohan, M. Muehlebach, T. Widmer and D’Andrea, “R.: The Cubli: A reaction wheel based 3D inverted pendulum,” 2013 European Control Conference (ECC), pp. 268-274. 2013.