1. ABSTRACT

We designed and built a nouvelle autonomous / semi-autonomous differential drive mobile robot to specifically collect up to 50 tennis balls. The nouvelle collection mechanism is based on a spinning drum that pushes tennis balls into a lightweight basket on top of the robot. The mobile robot has its own on-board intelligence where it is equipped with a myRIO embedded device. The robot can identify, localize, and collect tennis balls autonomously or, alternatively, a tennis player can command the robot to move to and collect balls through a mobile-phone application.

Keywords: mobile robot, tennis balls, collection, human-robot interaction, mechatronic design

2. INTRODUCTION

Tennis players need to collect the played balls from different locations in the court. In training sessions, many balls are used and played. This makes the collection a tedious and tiring process. Robert and Gwin [1] introduced a device for retrieving tennis balls. The device comprises an elongated handle member provided with hand grips at one end and a V-shaped collection member at an opposite end, the device is manually used by human. On the other hand, Tsai [2] created a robot for collecting table tennis balls, it is configured for receiving the picked table tennis balls therein. The robot travels on the ground along a predetermined route. This paper proposes a solution to solve the tennis ball collecting problem by designing and building a nouvelle autonomous / semi-autonomous mobile robot to specifically collect up to fifty tennis balls. The conceptual design of the nouvelle robot together with the selection of materials and components are detailed in Section 3. In Section 4, the kinematic and dynamic models along with the constraints are derived whereas the motion planning problem is stated and solved in Section 5. Section 6 presents the three layers of the control system and Section 7 discusses implementation and interface programming issues.

3. CONCEPTUAL DESIGN
A working prototype of the proposed novel design is shown in Figure 1. It has a rotating drum, in front of the robot colored in blue, that pushes the balls through a tangential force to the storage area.

The proposed design is conceptualized to meet a set of requirements elicited through literature review and interviews with tennis players. The robot must be able to collect 50 balls within 5 minutes, the storage area must be detachable from the robot, the robot should have its onboard power supply, and finally the robot must be safe for users and people in the tennis field as well as it should be user friendly with convenient interaction modality.

Here, the tennis ball collecting robot is driven by two independent wheels actuated by two DC motors with an electrical driver and encoder for each of them. Four castors are used to support and stabilize the robot. The robot collection mechanism is based on a spinning drum that pushes tennis balls into a storing basket. The drum is actuated by a DC motor commanded by H-bridge driver. A battery is used to power the robot and a voltage regulator shield is used to protect sensitive parts. A National Instruments embedded system, myRIO, is used as an information processing unit [3]. The robot can collect tennis balls autonomously or alternatively, the human can command the robot through a mobile-phone application through Bluetooth to drive it toward the balls to collect them.

![Figure 1: The designed tennis-ball collecting robot](image)

The detailed design consists of five subsystems that are synergistically integrated. These subsystems are shown in Figure 2.
Mechanical design and material selection

The robot components are the mechanical frame, wheels, collecting mechanism and storage area. CATIA software [4] is used for parts design. SolidWorks software [5] is used for parts assembly whereas SolidThinking software [6] is used for rendering, material selection and load analysis.

The mechanical frame consists of two main parts: the body frame and the supporting frame which carries all mechanisms, mechanical, electrical parts and the collecting mechanism. In addition, a ramp (transparent in Figure 1) is mounted behind the drum with an angle of 35 degrees to guide the balls to storage area. Two types of casters are used. Back casters are used to enable relatively easy rolling movement and stability of the robot. Front casters are used to manually drag the robot like a travel bag. The collecting mechanism drum is manufactured from layers of compressed sponge covered with a layer of chamois leather to enhance friction and traction.

Material selection and load analysis

The maximum stress and maximum deflection due to the maximum possible load placed on the robot are found numerically by using SolidThinking software. Aluminum 6061 alloy is selected to be the metal for the frame as it has a light weight and high stiffness. Its cross section is a hollow rectangle of 90 cm by 45 cm with a thickness of 1 cm. Its yield strength is 56 MPa.
Plastic (acrylic) is selected for the storage area and the support frame. Its thickness is 1 cm. Its yield strength is 45 MPa. To find the maximum deflection of the mechanical structure, the maximum load will be applied in the worst scenario to cause the maximum stress. By applying 30 N vertically downwards at the storage area and 20 N vertically downwards at the support frame, the stress is found to be 3.797 MPa. The locations where the maximum stress occurs are shown in Figure 3.

![Figure 3: Location where the maximum stress occurs](image)

The results, as well, show that the maximum von Mises stress is 6.6661 MPa whereas the maximum bending stress in the Aluminum part is 56 MPa and that in the plastic (acrylic) parts is 45 MPa. This leads to the following minimum factors of safety:

\[
\eta_{Al} = \frac{\sigma_u}{\sigma_e} = \frac{56}{6.661} = 8.407, \\
\eta_{plastic} = \frac{\sigma_u}{\sigma_e} = \frac{45}{6.661} = 6.756,
\]

The deflection due to bending is shown in Figure 4, and the maximum deflection is 0.24 mm. This deflection is acceptable and ignorable.

![Figure 4: The maximum deflection](image)
4. KINEMATICS AND DYNAMICS OF THE MOBILE ROBOT

The proposed robot is considered as a Differential Drive Mobile Robot (DDMR), which is subject to the following non-holonomic constraints [7]:

1. Pure rolling constraint which means that each wheel must maintain only a one contact point (P) with the ground [8] as shown in Figure 5. That is:

\[ \dot{x} \sin \theta - \dot{y} \cos \theta = 0 \]  

(1)

Which can be represented in Pfaffian form [7] as:

\[
\begin{bmatrix}
\sin \theta & - \cos \theta & 0
\end{bmatrix} \dot{q} = 0
\]  

(2)

where \( q = [x \ y \ \theta]^T \) is the generalized coordinate vector. Equation 2 is not integrable causing the nature of the constraint to be non-holonomic.

![Figure 5: Generalized coordinates for a rolling wheel on a plane](image)

2. No lateral slip motions: The robot can move only in a curved motion (forward and backward) but not sideward. This means

\[ \dot{y}_r = 0 \]  

(3)

where \( \dot{y}_r \) is the robot velocity along y axis as shown in Figure 6.

**Kinematic model**

DDMR is considered as a unicycle vehicle that has a single orientable wheel [7]. Its configuration is completely described by \( q \), see Figure 6. The kinematic equations are derived for a point, \( C \), that lies halfway between the wheels as shown in Figure 6. It has two velocity components which are the translational velocity, \( v \), along the \( x_r \) axis and angular velocity, \( \omega \), around a perpendicular axis.

Taking the above-discussed constraints into account, the relationship between the geometric velocities, \( \dot{q} \), and the robot velocities is given as:
Furthermore, the robot velocities are related to the right and left wheel rotational velocities, \( \omega_R \) and \( \omega_L \), according to

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} =
\begin{bmatrix}
\cos \theta \\
\sin \theta \\
0
\end{bmatrix} v +
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} \omega
\]

(4)

Furthermore, the robot velocities are related to the right and left wheel rotational velocities, \( \omega_R \) and \( \omega_L \), according to

\[
v = \frac{R(\omega_R + \omega_L)}{2}, \quad \omega = \frac{R(\omega_R - \omega_L)}{2L}
\]

(5)

where \( R \) is the radius of the wheels and \( L \) is the distance between their centers.

**Dynamic model**

The dynamic model of the DDMR, which represents the relation between input torques and resulting robot accelerations, is essential for simulation, analysis of robot motion and for the design of motion control algorithms [9]. It can be derived following either the Lagrange or the Newton-Euler methods to be:

\[
\dot{v} = \frac{1}{mR}(\tau_R + \tau_L)
\]

(6)

\[
\dot{\omega} = \frac{L}{IR}(\tau_R - \tau_L)
\]

(7)

Where, \( \dot{v} \) is the robot translational acceleration, \( \dot{\omega} \) is the robot angular acceleration, \( m \) is the robot total mass, \( I \) is the moment of inertia of the robot about the vertical axis passing through point \( C \), \( (\tau_R, \tau_L) \) is the driving motor torques.

**5. MOTION PLANNING**
For the execution of a specific robot task, the mobile robot needs to translate from an initial posture $q_i = [x_i \ y_i \ \theta_i]^T$ to final posture $q_f = [x_f \ y_f \ \theta_f]^T$ in a specific time $(\Delta T)$. This is a nontrivial problem due to the holonomic constraints. Only two torque inputs, $\tau_r$ and $\tau_l$, are available. The resulting velocities are $v$ and $\omega$. The solution presented in [7] is adopted here.

**Path and timing law**

Assume that the robot plans a point to point trajectory $q(t)$, it can be broken down into a geometric path $q(s)$, where $q' \neq 0$ for any value of $s$, and a timing law where $\dot{s}(t) \geq 0$. The value of $s$ is the arc length along the path. Then the generalized velocity can be obtained as

$$\dot{q} = \frac{dq}{dt} = \frac{dq}{ds} \dot{s} = q' \dot{s} \tag{8}$$

where the prime symbol denotes differentiation with respect to $s$. Therefore, all the admissible paths for the unicycle can be formulated as

$$x' = \ddot{\vartheta} \cos \theta \tag{9}$$
$$y' = \ddot{\vartheta} \sin \theta \tag{10}$$
$$\theta' = \dddot{\vartheta} \tag{11}$$

Where the value of $\ddot{\vartheta}$, and $\dddot{\vartheta}$ are computed as

$$\ddot{\vartheta}(s) = \pm \sqrt{(x'(s))^2 + (y'(s))^2} \tag{12}$$
$$\dddot{\vartheta}(s) = \frac{y''(s)x'(s) - x''(s)y'(s)}{(x'(s))^2 + (y'(s))^2} \tag{13}$$

The velocities of the robot are then computed as

$$v(t) = \ddot{\vartheta}(s) \dot{s}(t) \tag{14}$$
$$\omega(t) = \dddot{\vartheta}(s) \dot{s}(t) \tag{15}$$

**Planning via Cartesian polynomials**

The planning problem can be solved by interpolating the initial values $x_i$, $y_i$ and the final values $x_f$, $y_f$ of the flat outputs $x, y$, by letting $s_i = 0$ and $s_f = 1$ and using the following cubic polynomials, that automatically satisfy the boundary conditions on $x$ and $y$,

$$x(s) = s^3 x_f - (s - 1)^3 x_i + \alpha_x s^2 (s - 1) + \beta_x s (s - 1)^2 \tag{16}$$
$$y(s) = s^3 y_f - (s - 1)^3 y_i + \alpha_y s^2 (s - 1) + \beta_y s (s - 1)^2 \tag{17}$$

The orientation at each point being related to $x', y'$ is computed as
\[ \theta(s) = a \tan^2(y'(s), x'(s)) + K\pi, \quad K = 0, 1 \]  
(18)

The values of \( \alpha_x, \alpha_y, \beta_x, \beta_y \) in equations 16 and 17 are computed as following

\[
\begin{bmatrix}
\alpha_x \\
\alpha_y
\end{bmatrix} = \begin{bmatrix}
k \cos \theta_f - 3x_f \\
k \sin \theta_f - 3y_f
\end{bmatrix} 
\]  
(19)

\[
\begin{bmatrix}
\beta_x \\
\beta_y
\end{bmatrix} = \begin{bmatrix}
k \cos \theta_i + 3x_i \\
k \sin \theta_i + 3y_i
\end{bmatrix} 
\]  
(20)

The choice of \( k \) has a precise influence on the obtained path.

**Planning results**

For all of the following maneuvers, the timing law \( s(t) \) is selected as a 5\(^{th}\) order polynomial with the following conditions, \( s_i = 0, \ s_f = 1, \dot{s}_i = 0, \dot{s}_f = 0, \ddot{s}_i = 0, \ddot{s}_f = 0 \) and \( \Delta T = 3 \text{ sec.} \) the obtained timing law \( s(t) \) is shown in Figure 7.

![Image](image.png)

**Figure 7: Timing law \( s(t) \)**

Different paths are produced by the planner shown in Figures 8 and 9. Figure 8 shows planning with \( q_i = [0 0 0]^T \) and \( q_f = [-1 -3 0]^T \), Figure 9 shows planning with \( q_i = [0 0 0]^T \) and \( q_f = [4 -3 0]^T \).

Planning a parallel parking maneuvers via cubic Cartesian polynomials are shown in Figures 10 and 11 with \( q_i = [0 0 0]^T \) and \( q_f = [0 4 0]^T \).

Planning a pure reorientation maneuvers via cubic Cartesian polynomials are shown in Figures 12 and 13 with \( q_i = [0 0 60]^T \) and \( q_f = [0 0 0]^T \).
As already noticed, the unicycle never inverts its motion, that is forward because $k > 0$. For $k < 0$, the maneuvers would have been performed in backward motion with different paths.
6. ROBOT CONTROL

The robot must be able to implement the trajectories generated by the planner. It should be able to overcome the effect of variable ground friction, external disturbances, and variations in wheel radius. In order to ensure that, three levels of control (inner loop, outer loop and human in the loop) are considered. The inner and the outer loops are built and implemented using MATLAB/Simulink software and the human in the loop is implemented by a mobile application. A decentralized control architecture with a PID controller is used for the inner loop to achieve the desired wheel speed (servo level) based on encoder readings. The outer loop is designed with a PI controller to ensure the desired orientation (without angular velocity drift) based on an angular velocity measurement through a gyroscope. Further a human commanding the robot can be considered as the core of a third loop.

Inner loop control design and simulation results

The decentralized controllers are implemented to control the speed (servo level). See Figure 14.

![Decentralized PID controller architecture for servo level](image)

Figure 14: Decentralized PID controller architecture for servo level

PID controller is designed based on tuning method. The PID control law is given as [11]:

\[
G_{PID} = K_p + \frac{K_i}{S} + K_dS
\]  (21)

\[
G_{PID} = 14.5 + \frac{118.75}{S} + 0.045
\]  (22)

Equation 21 shows the PID transfer function where \(K_p\), \(K_i\) and \(K_d\) are the proportional, integral and derivative gains respectively. Equation 22 shows the tuned PID transfer function.

The results of PID is shown in Figures 15 and 16.
Figure 15: Comparison between the desired and actual speed of right motor

Figure 16: Comparison between the desired and actual speed of left motor

Outer loop control design and simulation results
Outer loop controller is implemented to ensure the desired orientation (without drift). See Figure 17. A PI controller with feedforward is designed to make the error (difference between desired and actual angular velocities about the vertical axis) become zero. The controller computes a modified desired angular velocity to compensate for the error.

![PI controller architecture for outer loop](image)

Figure 17: PI controller architecture for outer loop

Gyroscope sensor signal is used to measure the actual robot angular velocity $\omega_{\text{real}}$. The PI control law is given as [11]:

$$ G_{PI} = K_p + \frac{K_i}{s} \quad (23) $$

$$ G_{PI} = 2.7 + \frac{1.3 \times 10^5}{s} \quad (24) $$

Equation 23 shows the PI transfer function, $K_p$ and $K_i$ are the proportional and integral gains respectively. Equation 24 shows the tuned PI transfer function. The robot drifts when one of its wheels differ in the diameter from the other. For purpose of simulation, right wheel diameter is changed to 95% of its real size. Figure 18 shows the very small error signal obtained by using the PI controller.

![Error signal for outer loop controller](image)

Figure 18: Error signal for outer loop controller

Figure 19 shows the robot path without controller and Figure 20 shows the PI controller result.
7. IMPLEMENTATION

The mobile robot presented in this paper is built and tested. Different controllers and routines are embedded on its myRIO board through the LabVIEW environment. Two operating modes are highlighted here.

Manual mode

This mode promotes a tennis player commanding the robot to move and collect the balls via a mobile-phone application. Due to this mode the robot acts as follows: When the manual mode has been selected, the Bluetooth module activates and the system waits for the user connection. After the connection, the system starts receiving the user commands. A case structure is used to control the system based on the received commands. An I2C technique is used for interfacing the system controller with the driving motors controller. A digital pulse with modulation technique is used for interfacing the system controller with the collecting mechanism motor driver. The system gets back to the default values when it turned off. Figure 21 shows the android mobile-phone application.

Autonomous/semi-autonomous mode
In this mode, the robot localizes the balls and accordingly the presented planning algorithm decides the robot motion. Figure 22 shows the LabVIEW front panel that implements a graphical user interface for the planning algorithm.

![LabVIEW front panel](image)

Figure 22: the graphical user interface of the planning algorithm

### 8. CONCLUSION AND FUTURE WORK

In this paper, we designed and built a novel autonomous/semi-autonomous mobile robot to specifically collect up to fifty tennis balls. The robot navigates through two independently-actuated wheels and is stabilized by casters. The novel collection mechanism is based on a spinning drum that pushes tennis balls into a lightweight basket on top of the robot. A motor makes the drum spin at the desired speed, that is experimentally verified. Three successive working prototypes have been designed, built and tested to make sure the adopted ideas and conceptual designs are applicable and robust.

The mobile robot has its own onboard intelligence where it is equipped with a myRIO embedded device. The robot can navigate from the initial posture to the final posture based on a programmed path planning algorithm or, alternatively, a tennis player can command the robot to move and collect balls through a mobile-phone application.

LabVIEW codes are executed on the robot impeded device and the robot is controlled by the mobile application and moves based on user commands and collects the balls when the manual mode is activated. In that case the robot is controlled through the autonomous/semi-autonomous mode, in which the robot is commanded based on the planning algorithm and moved from the initial posture to the final posture following the generated trajectory.

In the future, a global camera system can be added in each corner of the court, and a central image-processing system for identifying and localizing all tennis balls on the court and generating an optimal path plan. A local camera mounted on the robot to collect balls randomly from the court and may help the global camera to detect the position of the tennis ball with high accuracy. Furthermore, sonar sensor can be used for collision avoidance.
References


