XVII. reál- és humántudományi Erdélyi Tudományos Diákköri Konferencia (ETDK)
Kolozsvár, 2014. május 15–18.

ASURO robotok vezérlése okostelefonokról

Szerző:
Mátis Szilárd-Gábor
Babeş-Bolyai Tudományegyetem, Kolozsvár, Matematika és Informatika Kar,
informatika szak, III. év

Témavezetők:
dr. Simon Károly egyetemi adjunktus,
Babeş-Bolyai Tudományegyetem, Kolozsvár
Szélyes Levente CEO,
Codespring Kft.

Kivonat

A dolgozat az ASURO projektet mutatja be, valamint a projekt fejlesztése során felhasznált hardver és szoftver eszközöket, technológiákat és fejlesztési módszereket tárgyalja.

A projekt célja algoritmusok és szoftvermodulok kidolgozása Arexx robotok vezérlésére. A fejlesztési folyamat során két robottípushoz készültek el ilyen modulok, az ARX-03, valamint a kicsivel fejlettebb és magasabb szinten, Arduino függvények segítségével is programozható AAR-04 típusokhoz. A megvalósítás során az alapfelszereltségükön túl a robotok további hardver modulokkal is kiegészültek (pl. további szenzorok és kommunikációs modul).

Fontos része a projektnek az Android platformon futó mobilalkalmazás, amelynek segítségével a felhasználók mobiltelefonjuk képernyőjén láthatják a robotra szerelt kamera által közvetített képet, valamint a szenzorok által rögzített információkat és telefonjuk dőlésszögének változtatásával vagy a képernyőn megjelenített virtuális nyomógombok segítségével irányíthatják a robotot.

Tartalomjegyzék

Bevezető

A dolgozat az ASURO projektet mutatja be, valamint a projekt fejlesztése során felhasznált hardver és szoftver eszközöket, technológiákat és fejlesztési módszereket tárgyalja.

A dolgozatban két robot kerül bemutatásra, az ARX-03 ASURO, illetve az AAR-04 ASURINO robotok. Mindkét robot az Arexx cég által gyártott kis robotautó, kifejezetten oktatási célokra fejlesztve. A robotok alapfelszereltségéhez tartoznak különböző szenzorok (sebességmérő, vonalkövető, ütközés szenzorok). Alapfelszereltségén túl, a projekt keretein belül használt robotok kiegészítő modulokkal is rendelkeznek: egy Arexx ULTRASONIC SET és egy Arexx Minesweeper kit segítségével az alapfunkcionalitásokon kívül lehetőség van távolságfelmérésre és fémdetektálásra. A nagyobb távolságból való irányíthatóság érdekében a robotokra felkerült egy-egy bluetooth modul is.

A felsoroltakon kívül az ASURO projektben felhasznált ASURINO robotautóra felkerült egy Liquid Image Ego típusú WIFI kamera, amely HD minőségű képet képes rögzíteni, illetve kisebb felbontásban továbbítani.

A projekt elsődleges célja az, hogy a felhasználók mobiltelefonjuk képernyőjén láthassák az autóra szerelt kamera által továbbított képet, valamint a szenzorok által rögzített információkat és telefonjuk mozgatásával (dőlésszög változtatása) irányíthassák az autót.

A dolgozat első részében a felhasznált hardver eszközök (a robotok, az ezekre felszerelhető kiegészítő hardver eszközök, a kép rögzítésére és továbbítására szolgáló kamera) bemutatása kapott helyet. A második rész ismerteti a fejlesztés során használt módszereket, valamint a felhasznált fejlesztési eszközöket. A harmadik részben a felhasznált technológiákról esik szó, itt kerülnek bemutatásra az Arduino és az Android platformok. A negyedik rész az ASURO projekt követelményeit, a projekt szerkezetét, illetve megvalósításának fontosabb elemeit mutatja be. Az ötödik rész egy konkrét használati esetet tárgyal, majd a következtetések és továbbfejlesztési lehetőségek következnek.

A fejlesztés alatt használt technológiákat, módszereket a dolgozat szerzője a Codespring Kft szervezésében tartott Mentorprogram keretein belül sajátította el. A fejlesztési folyamat a cég megbízott szakembereinek szakmai irányításával, a cég infrastruktúráját igénybe véve történt.

1. Felhasznált hardver eszközök

Az ASURO projekt keretein belül használt ARX-03 és AAR-04 Arexx robotok mikrokontrollereken keresztül vezérelhetőek, alapfelszereltségükhöz tartozik több szenzor, és a projekt fejlesztése során további hardver eszközökkel is kiegészültek.

1.1. Mikrokontrollerek

A mikrokontroller, vagy mikrovezérlő tulajdonképpen egy kis számítógép, amely tartalmaz egy mikroprocesszort és több különböző perifériát, amelyeket egy lapkára integrálnak. Rendkívül elterjedtek, míg egy átlagos háztartásban 1-2 személyi számítógép található, mikrokontrollerekből 30-40 is előfordulhat, mivel a digitális eszközök többségében megtalálható legalább egy.

A mikrokontrollerek tartalmaznak minden szükséges erőforrást (processzor, input/output memória - flash vagy ROM memória, kisméretű általános célú memória - RAM) és perifériát (portok, timer-ek, analóg-digitális átalakító - ADC), amelyek szükségesek egy adott feladat elvégzéséhez. A kontrollerek a feladatuknak megfelelően változhatnak, egyesek egyszerű feladatok megoldására tervezettek, kis órajelen működnek, keveset fogyasztanak (például távirányítókban található kontrollerek), mások összetettebb feladatok elvégzésére is alkalmasak.

A programokat több módszerrel be lehet tölteni egy mikrovezérlő memóriájába. A memória közvetlen módon újraprogramozható egy SPI soros interfészen keresztül, vagy a chippen található indító program (bootloader) által. A bootloader előnye, hogy nincs feltétlenül szükség kábelekre egy újraprogramozáshoz, így kényelmesebben megoldható a feladat, viszont hátrány lehet, hogy egy adott méretű tárhelyet elfoglal a memóriából.

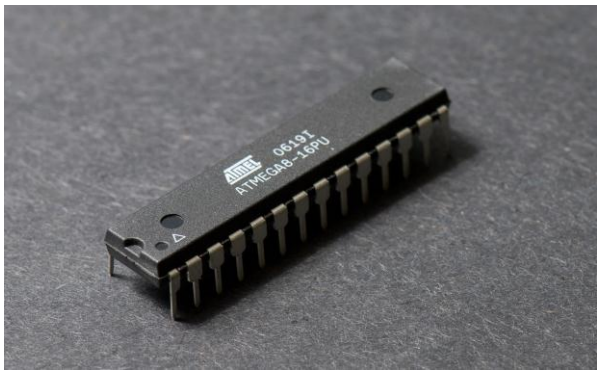
Az Arexx robotok központi alkatrésze szintén a mikrokontroller. Ezeknek típusa robotonként változhat, így esetenként a robotok képességei is változnak. Az ASURO ARX-03 típusú robot egy Atmel Atmega8 mikrokontrollerrel rendelkezik, míg az AAR-04 ASURO Arduino robot Atmel Atmega328P típusú vezérlést kapott.

1.1.1. Az Atmel Atmega8 mikrokontroller

Az Atmel cég által 1996-ban kifejlesztett ATmega8 egy Harvard-architektúrájú[1], 8 bites, RISC típusú mikrovezérlő, amely az AVR mikrokontroller családba tartozik. Az AVR

[1] A műveletek és adatok továbbítására használt útvonalak különválasztására irányuló alapelv

mikrokontrollereknél kezdtek el először chipre integrált flash memóriát használni a programtárolásra, az egyszer programozható ROM, EPROM és EEPROM memóriák helyett.

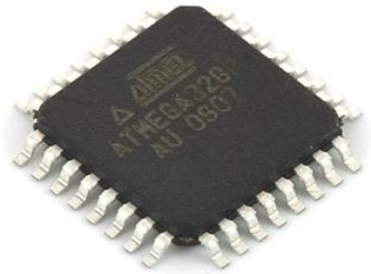


1. ***ábra*:** Atmega8-as mikrovezérlő

Az ATmega8 esetében a Flash, az EEPROM (512 byte) és az SRAM (1 kilobyte) egy chipre vannak integrálva, nincs szükség külső memóriára. A mikrokontroller nevében megjelenő 8-as érték utal arra, hogy 8 kilobyte SPI soros interfészen keresztül újraprogramozható flash memória biztosított a programok számára [1]. A mikrovezérlő biztosít továbbá 23 darab általános célú ki- és bemeneti csatornát és 32 darab általános célú regisztert.

1.1.2. Az Atmel Atmega328P mikrokontroller

Az ATmega8-as mikrokontrollerhez hasonlóan az ATmega328P is Harvard architektúrájú, 8 bites RISC mikrovezérlő. Maximálisan 20 MHz órajellel működő vezérlő, biztosít 32 Kbyte flash memóriát, 1024 byte EEPROM és 2 Kbyte RAM memóriát, 23 darab ki- és bemeneti portot, 32 darab 8 bites általános célú regisztert [2].



2. ábra: Atmel ATmega328P mikrokontroller

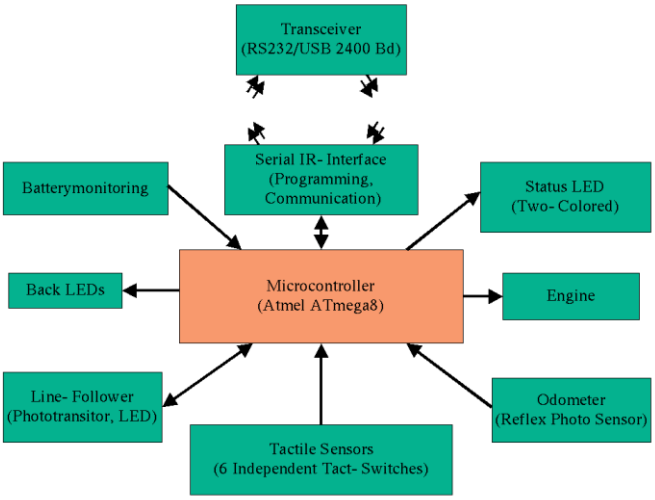
1.2. Az ASURO robot

Az ASURO egy kis robotautó, melyet eredetileg Robin Gruber és Jan Grewe fejlesztettek a német Légiközlekedési és Űrhajózási Központban, a Robotika és Mechatronika Intézetnél (German Aerospace Center - DLR, department for Robotics and Mechatronics). A robotot oktatási célokra készítették, elsődlegesen iskolai és egyetemi projektek számára.

Az ASURO ARX-03 típusú robot egy RISC-processzorral rendelkező Atmel Atmega8 mikrokontroller által vezérelhető. Alapfelszereltségéhez tartozik két egymástól függetlenül vezérelhető motor, hat ütközési szenzor, egy vonaldetektáló optikai egység és három kijelző LED. A felsoroltakon kívül további opcionális eszközök is felszerelhetőek a robotra (aknakereső, távolságmérő stb.).

3. ***ábra***: Az ASURO robot és felépítését mutató ábra

Az alapváltozat infraporton keresztül programozható és irányítható. A dolgozatban bemutatott projekt esetében, a nagyobb távolságból való irányíthatóság érdekében a robotra felkerült egy bluetooth modul is, amellyel 4-5 méteres távolságból is irányíthatóvá vált a robot.



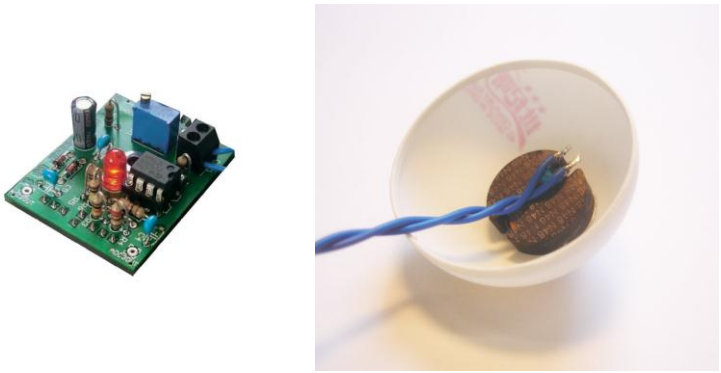
4. ***ábra***: Az ASURO felépítését mutató ábra (forrás: ASURO robot hivatalos dokumentációja [3])

A projekt keretein belül használt robot további kiegészítő modulokkal is rendelkezik: egy Arexx ULTRASONIC SET és egy Arexx Minesweeper kit segítségével az alapfunkcionalitásokon kívül képes távolságfelmérésre és fémdetektálásra. A hardver megszorításai miatt a fémérzékelő felszerelésével viszont a vonalkövető szenzor nem használható.

A robot programozása C nyelvben lehetséges. A kód lefordítására a Gnu C-Compiler fordító használható, egy ingyenes program, amely jól optimalizált kódot generál az ASURO Atmega8-as mikrokontrollere számára. A robotra a programokat infraporton keresztül tölthetjük fel, az *ASUROFlashTool* alkalmazás segítségével.

1.2.1. Kiegészítő alkatrészek

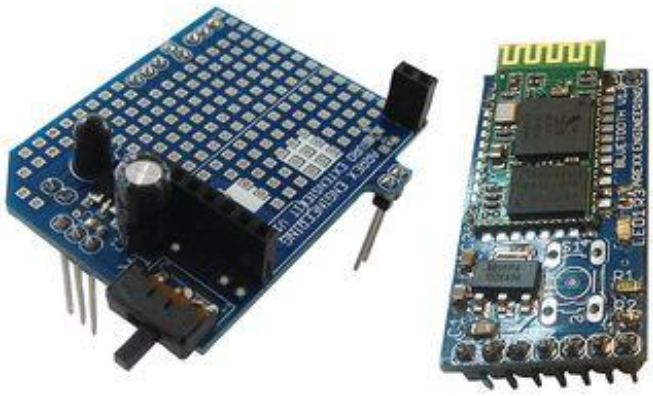
A robotra felkerült egy Arexx Minesweeper kit, amelynek segítségével a robot képessé vált fémdetektálásra. A modulhoz tartozik egy feszültségváltozás figyelésére használható tekercs, illetve a feldolgozást végző kiegészítő board. A fémkeresőhöz tartozó tekercs a robot alján elhelyezkedő asztalitenisz labdában kapott helyet, a kiegészítő tábla a robot tetején.



5. ábra: Arexx Minesweeper kit

Az Arexx Engineering által fejlesztett bluetooth modul használata révén a projektben használt robot nagyobb távolságból irányítható. A bluetooth kapcsolaton keresztüli irányításhoz szükséges lépések:

- A kliens által használt irányító eszközt (okostelefon, bluetooth kapcsolatot támogató számítógép stb.) társítani kell az ASURO roboton található bluetooth modullal.
- A bluetooth modulon található LED folyamatos kék villanásokkal jelzi, hogy a kliens eszköz csatlakoztatható a robothoz. Sikeres csatlakozáskor a LED villanásokból folyamatos világításba vált át.



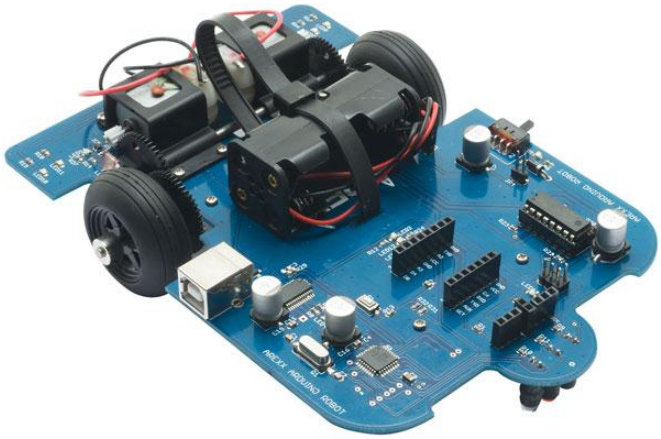
6. ábra: Arexx kiegészítő board és Bluetooth modul

A robotra felkerült az Arexx Engineering ARX-ULT10 Ultrasonic modul, amely segítségével a robot képes az előtte elhelyezkedő objektumok detektálására, illetve meg tudja becsülni távolságát ezektől.

7. ábra: A robotra felszerelt ultrahangos modul

1.3. Az ASURINO robot

Az ASURINO robot az Arexx cég által gyártott Arduino platformmal ellátott robot, hivatalos megnevezése Arexx Arduino Robot [4]. Az Arduino platform által a robot könnyebben programozható, a platform biztosít alapkönyvtárakat, amelyek hatékonyabbá teszik a programozást. A robottal alapértelmezetten USB porton keresztül lehet kommunikálni, de kompatibilis az ASURO kiegészítő modulokkal, így bluetooth kapcsolaton keresztül is irányítható.



8. ábra: Az ASURINO robot

A robot magját egy Atmega328P mikrokontroller képezi. Alapfelszereltségéhez tartozik két sebességmérő (wheel-sensors) optikai kapu, amely egy fototranzisztorból és egy infravörös LED diódából áll. Ezek sokkal pontosabb méréseket tudnak végezni, mint az ASURO robotnál használt sebességmérők, mivel itt nem a visszavert fénysugarakat figyeli a szenzor, hanem a kereket hajtó fogaskeréken levő lyukakon átszűrődő fény vezérli az optikai kaput. A sebességmérők mellett a robot rendelkezik még egy vonalkövető szenzorral, azonban nem rendelkezik az ASURO roboton megtalálható ütközési szenzorokkal. A kompatibilitásnak köszönhetően, az ASURINO robot felszerelhető ugyanazokkal a kiegészítő modulokkal, mint az ASURO robot.

A projektben használt robot esetében az alapértelmezett AAA típusú táp lecserélésre került AA típusú ellátásra, a korábban az ASURO robotnál tapasztalt áramellátási problémák miatt (folyamatos bluetooth kapcsolati probléma).

1.4. EGO kamera

Az ASURO projekt keretein belül használt Liquid Image által gyárott Model 727 típusú EGO Wifi kamera képes képek és HD videók rögzítésére, emellett az általa vett képet Wifi kapcsolaton keresztül továbbíthatja. Megjegyzendő, hogy egyszerre nem tud felvételt készíteni és képet továbbítani.

2. Fejlesztési módszerek és eszközök

Az ASURO projekt fejlesztése Scrum fejlesztési módszerből átvett elemek segítségével történt. A fejlesztés elején elkészült a termék backlog, egy user story lista, mely tartalmazta a fontosabb funkcionalitásokat. Ezt a listát a fejlesztés alatt a product owner (a szerepet a szakmai irányító töltötte be) módosította. A fejlesztés több iterációban, úgynevezett sprint-ekben történt, melynek során jellemzőek voltak a rendszeres találkozók.

A projekt forráskódján belüli változások nyomon követése a Mercurial osztott verziókövető rendszer segítségével valósult meg, amely a központi tároló (repository) mellett lokális tárolót is biztosít. A kényelmesebb kezelésben a Tortoise HG grafikus felhasználói felülettel rendelkező asztali alkalmazás segített, a központi tároló menedzsmentje a RhodeCode rendszer segítségével történt.

A fejlesztés alatt projektmenedzsment és hibakövető rendszerként a Redmine szoftver szolgált, amely egy webes kezelőfelületet biztosít a fejlesztők számára, egyszerűvé téve a feladatok beosztását és nyomon követését. A hatékony információ-megosztás érdekében a projekttel kapcsolatos dokumentumok és információk XWiki rendszeren belül voltak tárolva.

Az Android kliens fejlesztésére az Eclipse fejlesztői környezet, illetve az Eclipse mellé telepíthető Android ADT plugin volt használva.

A robotok szoftverének fejlesztése a Visual Studio-ra épülő Atmel Studio 6-os verziójának segítségével valósult meg. Az ASURINO robot esetében a fejlesztésre az Atmel Studio-hoz letölthető Arduino IDE plugin, valamint a platform által biztosított Arduino fejlesztői környezetek szolgáltak. A megírt programokat az ASURO robotra az robot gyártója által biztosított ASURO Flash segítségével történt infra porton keresztül, az ASURINO robot esetében az Arduino fejlesztői környezet által, USB-n keresztül kerültek feltöltésre a programok.

3. Felhasznált technológiák

3.1. ASURO robotok vezérlésére használt nyelvek és technológiák

Az ASURO robot szoftverének fejlesztése C nyelvben történt. Az ASURO robot gyártója alapértelmezetten biztosít egy ASURO függvénykönyvtárat, mely a legalapvetőbb funkcionalitásokat ellátja (kezdeti beállítások, motorokat és szenzorokat kezelő függvények, kommunikációt végző függvények). A projekt esetében bizonyos függvényeket módosítani kellett a megfelelő működés érdekében. Az alap függvénykönyvtárat szükséges volt kiegészíteni saját függvénykönyvtárakkal is.

3.1.1. Az Arduino platform

Az Arduino egy nyílt forráskódú, szabadon használható elektronikai fejlesztőplatform. A nevét egy ivreai uralkodóról kapta, kezdetben Arduin of Ivrea volt a fejlesztés neve, később lerövidült Arduino-ra.

A platform erőssége, hogy létrehoz egy felsőbb absztrakciós réteget, ezzel megkönnyítve a fejlesztést. Biztosít bizonyos általános műveleteket, amelyek különböző board-okon ugyanúgy működnek. A fejlesztéshez két fontos dologra van szükség: az Arduino Board-okra és az integrált fejlesztői környezetre.

Az Arduino Board egy elektronikai fejlesztőeszköz, segítségével az általunk megírt programokat futtathatjuk. Több típusú Arduino Board létezik, vannak egyszerűbbek és komplexebbek, művelet specifikusabbak. A felhasználók maguk is összeállíthatnak board-okat, vagy előre összeállított hardver eszközöket is beszerezhetnek. Lényegében mindegyik board egyedi, különböző méretűek és alakúak, különböző erőforrásokkal (ROM, RAM stb.) rendelkezhetnek. Az esetek többségében Atmel AVR mikrokontrollerekkel forgalmazzák ezeket, de található Intel processzorral ellátott board is. A Board-ok kiegészíthetőek különböző Shield-eknek nevezet modulokkal. A Shield-ek olyan elektronikai áramkörök, amelyekkel az Arduino rendszer az internetre, wifi hálózatra csatlakoztatható, vagy mechanikus elemek köthetőek hozzá (motorok, relék). A Shield-ek mellett általános kiegészítőkkel is ellátható a board, LCD kijelzővel, illetve USB soros adapterrel.

A felhasználók számára több fajta fejlesztői környezet, illetve kiegészítő plug-in áll rendelkezésre az Arduino platformra történő fejlesztéshez. A programok készítését egyszerűsített C++ nyelvben tehetik meg. Az Arduino platform egyben biztosít egy fejlesztői környezetet is, az Arduino IDE-t, amely egy platformfüggetlen, Java nyelvben megírt

környezet. Segítségével a programok fejleszthetőek, feltölthetőek az eszközökre és tesztelhetőek. Az Arduino IDE mellett letölthetőek más fejlesztői környezetekhez kiegészítő modulok: Arduino IDE Microsoft Visual Studio, Arduino IDE Atmel Studio, vagy Arduino Eclipse IDE.

Az Arduino az elmúlt évtizedben nagyon népszerűvé vált a fejlesztők körében, és a közösségnek létezik egy hivatalos internetes fóruma is, ahol megoszthatják észrevételeiket, tapasztalataikat, illetve forráskódjaikat.

3.2. Az Android platform

Az Android platform az utóbbi évek leggyakrabban használt mobil operációs rendszere. A platform fejlődése 2005-ben indult be (előtte is voltak fejlesztések, de nem értek el nagyobb sikereket), amikor a Google felvásárolta az *Android Incorporated* nevű céget. Az Android fejlesztéséért a Google által vezetett Open Handset Alliance nevű konzorcium felelős.

A platform sikerességét több tényezőnek köszönheti: látványos felhasználói felület, egyszerű használat, nyílt forráskód, könnyű fejleszthetőség, széleskörű kompatibilitás stb. Egy másik szempont az Androidot futtató készülékek fejlettsége, a hardverek minősége, ami egyrészt a gyors processzorokban és nagyméretű memóriában, másrészt a multimédia és kommunikációs eszközök sokaságában nyilvánul meg. Az Androidot már nemcsak az okostelefon- és tabletgyártók használják, hanem nagy érdeklődés mutatkozik iránta más ipari területekről is (a gépjárművek fedélzeti számítógépét, navigációs rendszerét gyártó cégek részéről, az ipari automatizálás irányából). A platform folyamatosan fejlődik, frissül, a legújabb verzió az Android 4.4 KitKat (a verziók különböző édességek neveit kapták).

Az Android alapját egy Linux-kernel képezi, ez legalsó szinten található, feladata biztosítani az alacsonyabb szintű rendszerfunkciókat, mint a memória kezelése, a folyamatok ütemezése, illetve ezen a szinten találhatóak a különböző hardver eszközök meghajtó programjai. A kernel felett található rétegben a különböző C/C++ programkönyvtárak helyezkednek el, részben ezekre épül a Dalvik virtuális gép (DVM). melynek feladata a Java-ban megírt alkalmazások futtatása. A legfelső réteg már teljesen Java alapú, itt találhatóak az alkalmazások, amelyeket a virtuális gép futtat. Ezek az alkalmazások a szintén Java alapú Application Framework réteg komponensein és szolgáltatásain keresztül kommunikálnak az alsóbb rétegekkel. Az alkalmazás keretrendszer feladata, hogy kiszolgálja az alkalmazásokat és hozzáférést biztosítson a rendszer különböző erőforrásaihoz.

3.3. Otto Event Bus

Az alsó rétegekben történő események a felsőbb rétegek fele történt továbbítása a Guava alapú, Android platformra fejlesztett Otto Event Bus segítségével valósult meg. Segítségével az objektumok feliratkozhatnak bizonyos típusú eseményekre (bluetooth kommunikációval kapcsolatos események, kamerával kapcsolatos események, stb.).

4. Az ASURO projekt

Az ASURO projekt az Arexx által gyártott ASURO és ASURINO robotokkal, a robotok szoftverének, valamint az Android platformon futó kliensalkalmazásnak a fejlesztésével foglalkozik. A projekt elsődleges célja az Arexx robotok irányítása az Android platformon futó mobilalkalmazás segítségével, valamint a robotokra szerelt kamera által továbbított kép és a szenzorok által rögzített információk megjelenítése az alkalmazást futtató készülék képernyőjén.

4.1. Követelmények és fontosabb funkcionalitások

Az ASURO projekt központi funkcionalitása a robotok irányítása, vezérlése. Ezt a felhasználók többféleképpen is megtehetik: mobiltelefonjuk segítségével, vagy egy asztali alkalmazáson (például HyperTerminal) keresztül, például a billentyűzet segítségével. A cél az, hogy bármilyen irányítási eszközt választ a felhasználó, a robot hasonlóan reagáljon, hasonlóan működjön.

4.1.1. A robot funkcionalitásai

a) Az irányítással kapcsolatos funkcionalitások:

- A robotnak több irányba is mozgathatónak kell lennie.
- A haladási sebességnek meghatározhatónak kell lennie.
- A robotnak fokozatosan kell változtatnia a sebességét, irányát, elkerülve a „rángatózó”, vagy „szögletes” mozgásokat. Például: a robot folyamatosan mozog előre *x* sebességgel, ezután kap egy megállj (vagy lassít parancsot). Ezt nem hajtja végre azonnal, hanem fokozatosan csökkenti sebességét, míg el nem éri a kívánt sebességet. Hasonlóan a gyorsítás esetében is, a robot fokozatosan növeli sebességét az elvárt sebességi fokozatig. Az oldalirányú mozgatásnál is hasonló a helyzet. Irányváltás esetén nem vált azonnal irányt, hanem lépésekben áll át az új helyzetbe. Például: a robot bizonyos szögben halad. Ha kap egy „előre” parancsot, akkor nem vált át azonnal egyenes irányú mozgásba, hanem fokozatosan csökkenti a szöget, és lassan beáll a kívánt helyzetbe.

b) A robot szenzorjaival kapcsolatos funkcionalitások:

A robotnak jeleznie kell a felhasználónak a szenzorokkal kapcsolatos eseményeket (ütközési szenzorok jelzései, akadály jelzése a távolságmérő szenzor által, fémdetektáló szenzorral kapcsolatos események).

Bizonyos szenzorokkal kapcsolatos eseményekre a robotnak intelligens módon kell reagálnia:

- Ütközés esetén a robotnak meg kell állnia, nem folytathatja tovább mozgását a korábban kijelölt útvonalon. A későbbiekben az irányító szoftver továbbfejleszthető, a robot az akadály pozíciójának függvényében megpróbálhatja kikerülni az ütközést okozó tárgyat.
- A távolságmérő szenzor esetében három távolsági szint különíthető el. Az első két szint esetében (a zöld és sárga szintek) a robot még csak jelzi a felhasználónak, hogy egy tárgyat detektál maga előtt. A harmadik (piros) távolsági szint elérésekor megszakítja mozgását, a továbbiakban nem engedélyezi az előre irányuló mozgatásokat, mindaddig, amíg fennáll az ütközés veszélye, a piros távolsági szint.
- A fémdetektáló szenzor által érzékelt eseményeket kell jeleznie a felhasználónak.

4.1.2. A kamerával kapcsolatos funkcionalitások

A kamerának a roboton kell elhelyezkednie és az általa vett képet továbbítania kell Wifi Hotspot-on keresztül. A továbbított képeket az Android platformon futó kliensalkalmazás jeleníti meg. A szenzorok által továbbított információk, egyfajta kiterjesztett valóságként (Augmented Reality) a kamera nézetben, a közvetített képen jelennek meg.

4.1.3. Az Android platformon futó alkalmazás követelményei

a) Az Android alkalmazás funkcionális követelményei:

A felhasználók az alkalmazás segítségével irányíthatják a robotot, bluetooth kapcsolaton keresztül. Az irányítási lehetőség kiválasztása során az alkalmazásnak automatikusan kell kapcsolódnia a robothoz, illetve a kamerához. Bármilyen kapcsolódási hibát jeleznie kell a felhasználónak. A kamerához történő kapcsolódás során fellépő hiba esetében, a felhasználó választhat, hogy ellenőrzi-e a wifi kapcsolatot, vagy kamera nélkül irányítja a robotot. Ha a robothoz történő kapcsolódás esetében lép fel hiba, az alkalmazás figyelmezteti a felhasználót, majd visszatér a főoldalra.

A felhasználónak lehetősége van több irányítási mód közül választani: a telefon dőltésével történő irányítás, vagy a telefon képernyőjére elhelyezett virtuális nyomógombok segítségével történő irányítás. Ugyanígy választhat kamerás, illetve kamera nélküli irányítási

lehetőségek közül is. Ezeket az opciókat egy beállítások nézetből tudja módosítani. Az irányítás közben a telefon képernyőjén láthatóak a robot szenzorjainak állapotkijelzői.

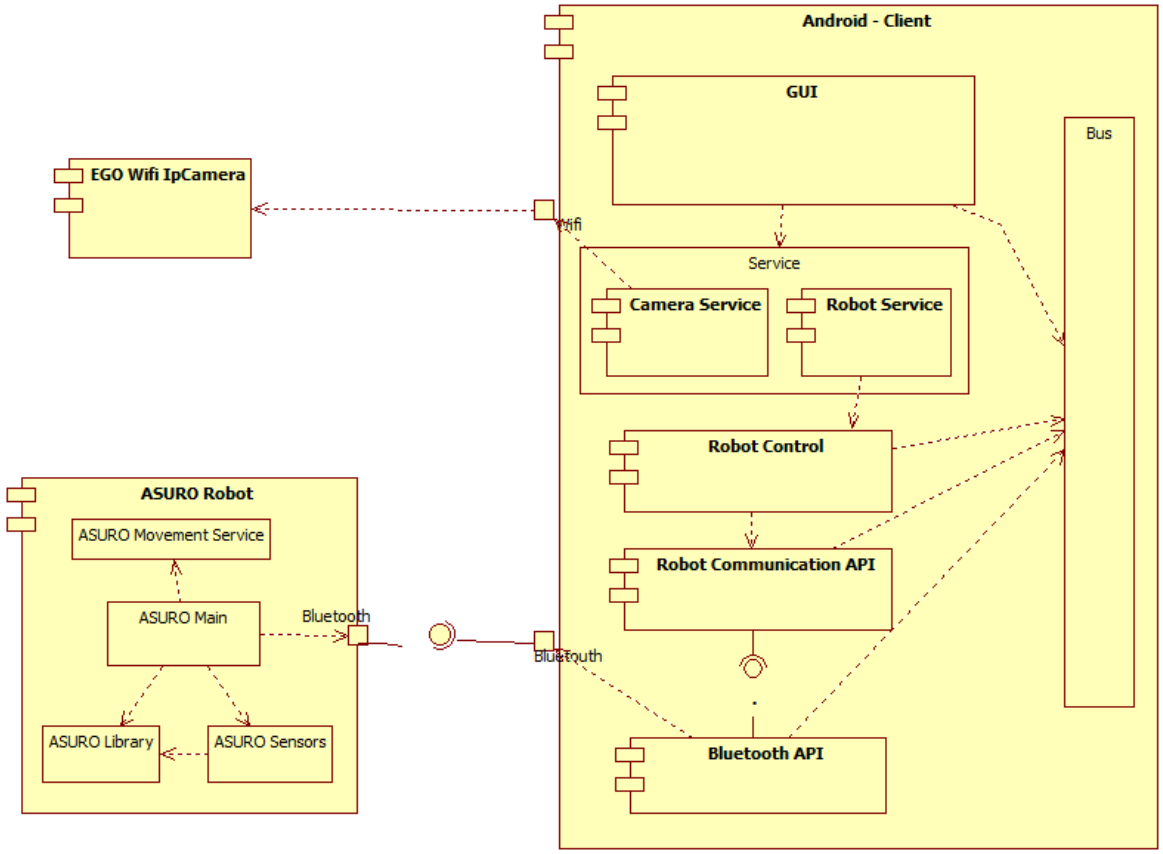
Az alkalmazásnak az általa ismert robotokat dinamikusan kell kezelnie, a felhasználónak lehetőséget adva új robotok hozzáadására vagy már meglévő robotok törlésére.

b) Az Android alkalmazás nem funkcionális követelményei:

Az alkalmazásnak felhasználóbarátnak kell lennie (például a fontosabb funkciókat egyetlen képernyőérintéssel el lehessen érni). Az alkalmazást futtató telefonnak támogatnia kell a bluetooth-, illetve a wifis kapcsolatokat, illetve minimum 2.3-as, vagy annál újabb Android verzióval kell rendelkeznie.

4.2. Architektúra

Az 9. ábrán látható az ASURO projekt architektúrája. A rendszer két fő komponensből áll: az ASURO robotra telepített modulból, illetve a robotot irányító, Android platformon futó mobilalkalmazásból. A rendszer kommunikál továbbá egy EGO wifi kamerával, amelyet a robotra lehet felszerelni, és így képet továbbítani általa a mobilalkalmazásnak.



9. ábra: az ASURO projekt architektúrája

Az ASURO robotra a fejlesztés C nyelvben történt, az alkalmazás több komponensből áll. Az *ASURO Library* a legalsó és legalapvetőbb réteg, a robot gyártója által biztosított standard könyvtár, erre a könyvtárra épülnek az *ASURO Sensors* és az *ASURO Main* rétegek. Az ASURO Library biztosítja az alapvető funkcionalitásokat, a robot mozgatási parancsait, a robot alap szenzorjaihoz tartozó, illetve a kommunikációhoz használt függvények. Az ASURO Sensors az ASURO robot szenzorjaival végezhető műveleteket biztosítja. Az alapvető könyvtár által biztosított szenzor műveleteket kibővíti, illetve itt találhatóak meg a kiegészítő szenzorokkal (ultrahangos távolságmérő, fémdetektáló) kapcsolatos függvények is. Az ASURO Movement Service rétegben a robot irányításához tartozó műveletek vannak csoportosítva. Ebben a rétegben történik a robot irányítási üzeneteinek a dekódolása, valamint az üzeneteknek megfelelő sebességek és irányok meghatározása.

Az ASURO Main képezi a robot vezérlési programjának központi elemét, ez a rész tartalmazza a főprogramot és itt találhatóak a robot alapbeállításai. Az ASURO Main fogja össze a különböző modulokat, ez a réteg ütemezi a robot irányításával kapcsolatos műveleteket, illetve a szenzorokkal kapcsolatos műveleteket.

A másik lényeges komponense a projektnek az Android kliens. Ezen belül több réteget különíthetünk el. Egyesek Android specifikusak, mások függetlenek az Android platformtól, így az utóbbiak más Java projektekben újrafelhasználhatóak. A legfelső szint, amellyel a felhasználó közvetlen interakciót valósít meg, a grafikus felhasználói felület (GUI). A GUI felelős a különböző nézetek megjelenítéséért, az ezek között történő váltásokért, a felhasználói események megfelelő alsóbb réteghez történő továbbításáért. Másik fontos szerepe a GUI-nak, hogy az alsóbb rétegekben történő eseményeket jelezze a felhasználó számára (például ha megszakad a kapcsolat a robottal vagy a kamerával). Az alsóbb rétegekben bekövetkező események felsőbb rétegekhez történő továbbításáért a Google Guava-ra épülő, Android specifikus Otto Event Bus felelős. A Bus működési elve egyszerű, publish-subscribe mintára épül. Az objektumok feliratkoznak a Bus-hoz (minden feliratkozó ugyanahhoz a Bus objektumhoz iratkozik fel, ezért a BusProvider segédosztály felelős).

A GUI alatt a szolgáltatási (Service) réteg helyezkedik el, amely két különböző részből áll: a Camera Service-ből, illetve a Robot Service-ből. A Camera Service egy interfészt biztosít, amelyen keresztül konkrét kameratípustól függetlenül elérhetővé teszi a szolgáltatásait. Szerepe a kapcsolódás a kamerához. A Robot Service szerepe a GUI-tól érkező, gyorsulásmérő szenzorral (Accelerometer) kapcsolatos események következményeként hívott műveletek végrehajtása. A Service továbbítja a különböző eseményekkel kapcsolatos üzeneteket a GUI és a Robot Control modul fele. Az alkalmazás szolgáltatási rétege platformfüggő, Android rendszeren használható.

A Robot Control modulon belül történik a mozgatásnak megfelelő irány meghatározása. A modul feladata irányítási parancsok küldése a robothoz, illetve a szenzorokra vonatkozó információk feldolgozása, és azok továbbítása a Bus segítségével.

A Robot Communication API a robot és az Android kliens közötti bluetooth kapcsolat létrehozásáért és kezeléséért felelős. Ehhez a Bluetooth API szolgáltatásit használja. A Bluetooth API biztosítja a kapcsolat létrehozásáért felelős osztályokat, illetve az üzenetek ki- és beolvasása is itt történik.

4.3. Megvalósítás rövid összefoglalása

Az ASURO robotok szoftvere és az Android kliens fejlesztése párhuzamosan történt. Mindkét esetben több lépésben haladt a fejlesztés.

4.3.1. Az ASURO robotok fejlesztése

A robotok fejlesztése során két lényeges rész különült el: egy megfelelő kommunikációs protokoll megalkotása, valamint a robotok irányításáért felelős modul fejlesztése.

Az fejlesztés során lényeges szerepet kapott a kommunikációs protokoll kialakítása, mely leírja, hogy a robot milyen üzeneteket fogad és küld. A robot több fajta kliens (mobil alkalmazás, asztali Hyperterminal) által is irányítható. A klienseknek ismerniük kell a robot protokollját, hogy kommunikálni tudjanak vele, a robotot működésbe hozhassák. A protokoll meghatározásában lényeges szempont az egyszerűség, a minél hatékonyabb üzenetváltás, az üzenetek könnyű ki- és becsomagolásának biztosítása. Kétirányú kommunikációról lévén szó, a robot mozgatási parancsokat, illetve a szenzor információkat kódoló üzeneteket cserél a csatlakozott klienssel. A protokoll egyszerű, egy byte hosszúságú karaktereket használ, ezáltal az üzenetek kisméretűek és asztali alkalmazásról (WASD billentyűk) is könnyen irányíthatóvá válik a robot.

Az ASURO robot irányítására szolgáló modul fejlesztése több iterációt igényelt. A különböző problémák megoldása és a vezérlés finomítása lépésekben történt.

Az első lépés az ASURO roboton található szenzorok tesztelése és működésük megértése volt. Minden szenzorhoz egy külön tesztalkalmazás készült.

A második lépésben történt meg a robot irányításának tesztelése. Az egyszerű mozgatások (előre-hátra, jobbra-balra, illetve ezek kombinációja) voltak először tesztelve. A tesztalkalmazás a motorokat irányította, minden lehetséges irányban mozgatva a robotot.

A harmadik iteráció keretein belül, a robot bluetooth kapcsolaton keresztüli irányításának teszteléséhez először a HyperTerminal nevű program volt felhasználva. A tesztprogram irányíthatóvá tette a robotot a billentyűzet segítségével.

A tesztelés alatt a bluetooth kommunikációval kapcsolatos probléma állt elő: rövid időközönként megszakadt a kapcsolat a robot és az irányító számítógép között. A gondot csak a robot és/vagy a gépen futó HyperTerminal újraindítása oldotta meg. A probléma forrása először nem volt nyilvánvaló, több hibalehetőség is felmerült. A jelenséget okozhatta az aszinkron üzenetküldéssel kapcsolatos szoftverhiba, vagy hardver hiba, például a bluetooth modul érzékenysége a feszültségingadozásra.

Egy átfogó tesztelés és hibajavítási fázis során először a ki- és bemenő üzenetek küldésének ellenőrzése történt meg. Külön ellenőrizve volt a kommunikáció csak bejövő, illetve csak kimenő üzenetekre, ezek kombinációjára, valamint üzenetküldésre és fogadásra a motorok működése közben. Az utolsó pont kivételével nem lépett fel az említett kommunikációs hiba, így nyilvánvalóvá vált, hogy azt nem szoftver probléma okozta, hanem hardverrel kapcsolatos gond: a modul túlérzékenysége a feszültségingadozásra.

A hiba kiküszöbölésének érdekében először az akkumulátorok lecserélése történt meg nagyobb kapacitásúakra. A csere csak minimális szinten javított a problémán, így is előfordult, hogy megszakadt a kapcsolat a robot és az irányító számítógép között. Következő lépésként külön lett választva a robot bluetooth moduljának, illetve magának a robotnak az energiaellátása, egy újabb akkumulátor került fel a robotra és ez a megoldás már sikeresnek bizonyult.

4.3.2. Az Android mobilalkalmazás fejlesztése

A mobilalkalmazás fejlesztése több lépésben történt, minden iterációban egy-egy réteg elkészítése volt a cél.

Az első iteráció célja a kamera által vett kép megjelenítése volt a mobil eszköz képernyőjén. A fejlesztés elején nem volt ismeretes a kamera által biztosított stream típusa, sem az URL, melyen keresztül a képet közvetítette. A kamera forgalmazója nem közölte a fenti információkat a kamerához tartozó dokumentációban, így egy harmadik fél által biztosított és a kamera gyártója által ajánlott Action Connect mobilalkalmazás log állományának vizsgálatából kellett a szükséges információkat kifejteni. A megfelelő információk ismeretében az Android platform által a megjelenítés már nem okozott problémát. Egyetlen szépséghiba lépett fel, hogy a kép pár másodperces késéssel jelenik meg csak a mobileszköz képernyőjén.

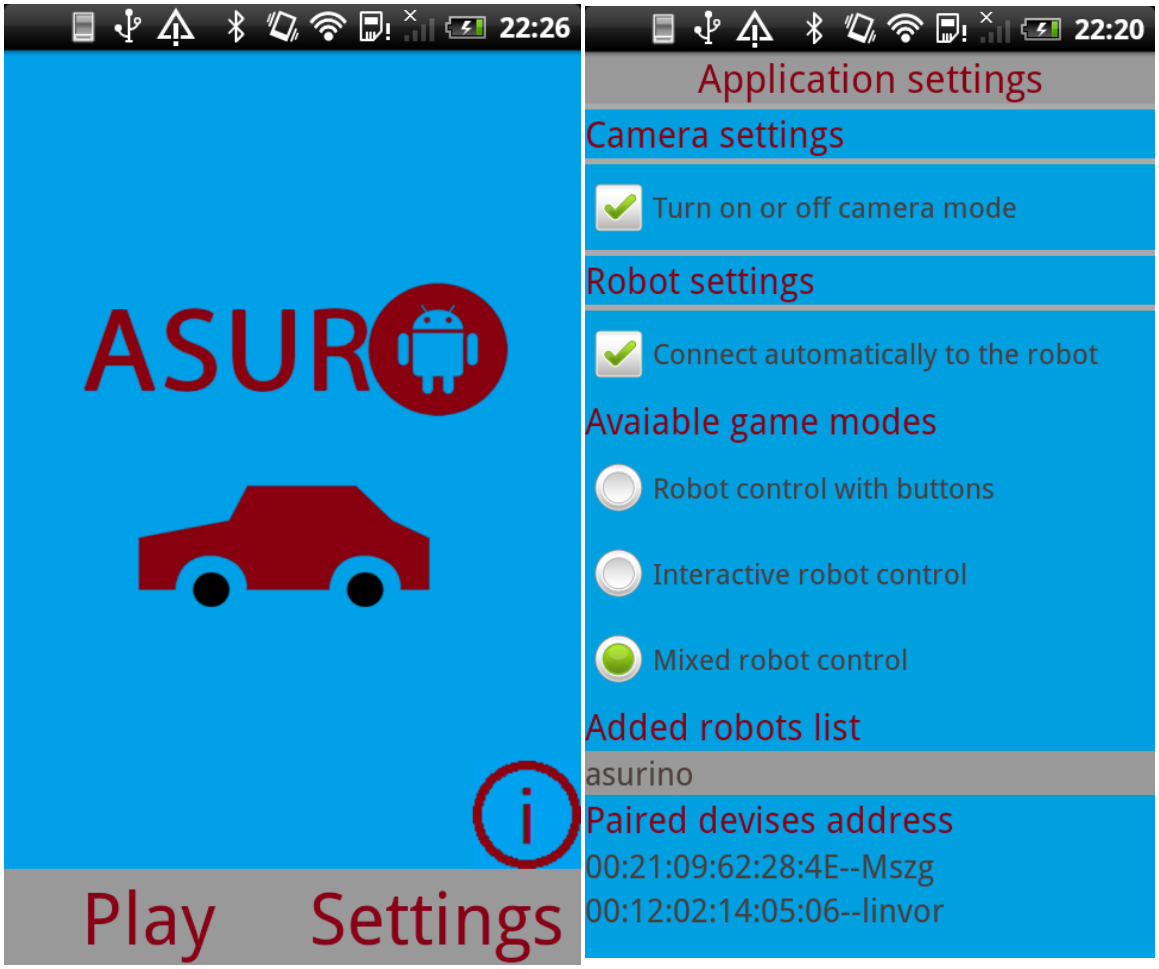
A második lépésben a robot irányításához szükséges modulok fejlesztése indult el, esetenként párhuzamosan. Egyrészt az alkalmazás által biztosítandó irányítási módok felépítése a grafikus felületen, másrészt az ezzel párhuzamosan a GUI-tól kapott információk feldolgozása a Robot Service által.

A harmadik lépésben a robottal történő kommunikáció és irányításért felelős modul megvalósítása készült el. A kapcsolatért és irányításért felelős rétegek megalkotásánál a cél az volt, hogy a modulok ne legyenek Android platformtól függőek, hogy ezek újrahasznosíthatóak legyenek más Java alkalmazások esetében.

A mélyebb rétegekben bekövetkezett események (üzenetérkezés, bluetooth kapcsolati hiba) a megfelelő felsőbb rétegekhez történő továbbítása fontos szerepet kapott. Ki kellett küszöbölni azt, hogy az alsóbb rétegekben létrejött eseményeknek minden rétegen keresztül kelljen menniük. A megoldást a Guava-ra épülő Android specifikus Otto Event Bus jelentette. Segítségével az objektumok biztosíthatnak, illetve feliratkozhatnak különböző esemény típusokra, valamint megfelelően kezelhetik ezeket.

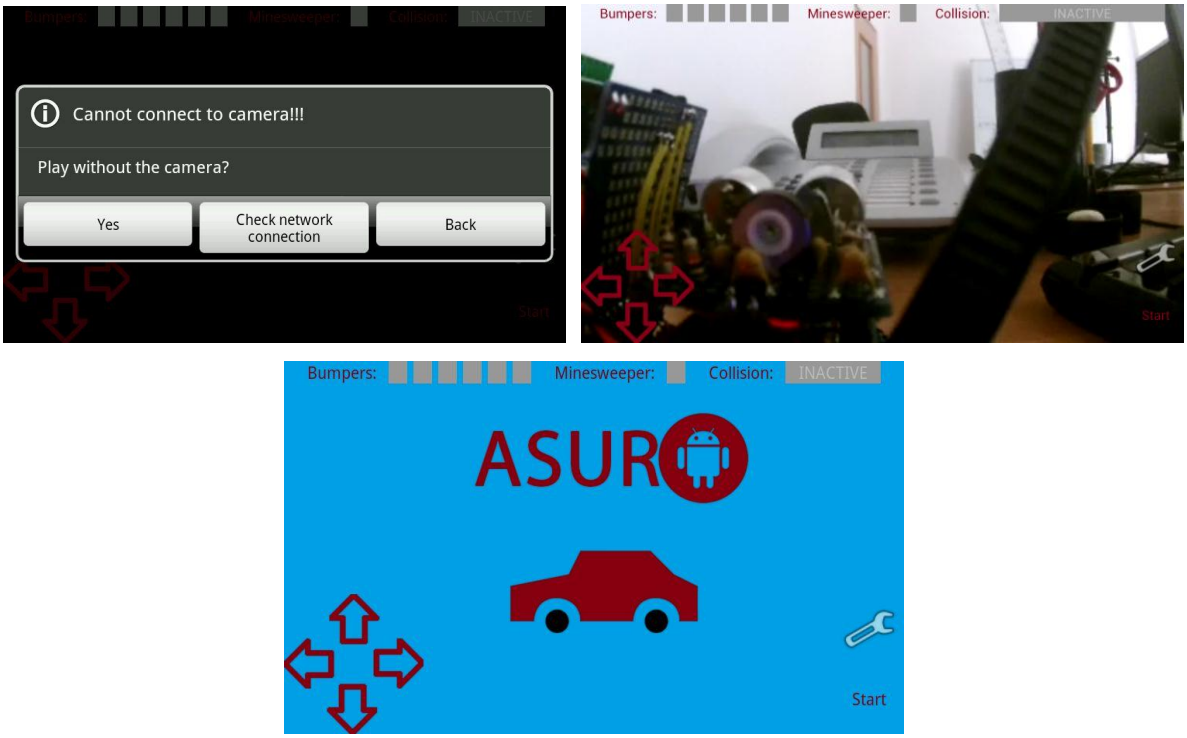
5. Az ASURO alkalmazás használata

A felhasználók az okostelefonjaik által irányíthatják a robotokat. Az alkalmazás elindításával a szoftver automatikusan próbál csatlakozni a beállított robothoz. Abban az esetben, ha a robot nem elérhető vagy nem volt beállítva alapértelmezett robot, akkor az alkalmazás hibaüzenettel értesíti a felhasználót a fellépő problémáról. Ha nem sikerül kapcsolódni a robothoz, akkor a felhasználó nem tud vezérlési nézetbe váltani.



10. ábra: Az ASURO alkalmazás főnézete és a beállítások nézet

A beállítások nézetben a felhasználónak több opció kiválasztására van lehetősége. Választhat kamera és kamera nélküli módok közül, beállíthatja a játékmódot (gombokkal, dőltéssel történő irányítási módok), az automatikus kapcsolódás lehetőségét, valamint a felhasználó hozzáadhat vagy törölhet robotokat, azaz a robotokhoz tartozó bluetooth modulok azonosítóit. Az első indításnál a felhasználónak kell hozzáadnia az alkalmazáshoz azokat a robotokat, amelyeket irányítani szeretne. Ehhez a felhasználónak ismernie kell a robotra felszerelt bluetooth modul azonosítóját. Ezeket az azonosítókat abban az esetben láthatja a felhasználó, ha készüléke párosítva volt a robotokkal. A felhasználónak lehetősége van ezeket az adatokat törölni is.



11. ábra: Az ASURO alkalmazás vezérlési nézetének állapotai: kamera hiba fellépése, irányítás kamera módban, irányítás kamera mód nélkül

A vezérlés elindításakor a megfelelő beállítások alapján felépül a nézet. Kamera nélküli mód esetében egy egyszerű háttérkép jelenik meg, kamera nézet esetében a készüléken megjelenik a közvetített kép (ha a korábbiakban a felhasználó csatlakoztatta a készülékét a kamerához). Problémák fellépése esetén a felhasználót értesíti az alkalmazás, több opciót kínálva neki megoldásként.

Következtetések és továbbfejlesztési lehetőségek

A projekt keretein belül sikerült megvalósítani a kiválasztott Arexx robotok irányítását és a robotokon található szenzor információk feldolgozását, valamint a kamera által vett kép fogadását a mobil kliensalkalmazáson belül. A fejlesztés során több probléma is jelentkezett, egyesek komolyabbak (pl. kommunikációs problémák), mások kevésbé befolyásolták a működést (pl. kamera kép megjelenítésének késése). A hibákat figyelembe véve és további ötletek alapján több továbbfejlesztési lehetőség is felmerült:

- A projekt kiegészíthető AI modulokkal. A robotok bizonyos helyzetekre automatikusan reagálhatnának, a felhasználói interakció igénybevétele nélkül. Egyik lehetőség az ultrahangos távolságmérő alapján történő ütközés-megelőzés, észlelt tárgyak kikerülése. Egy másik lehetőség a kamera által vett kép szerinti tájékozódás, tárgyak felismerése, észlelése, és ezek kikerülése ütközés esetén. Ennek megvalósításához a kamera kép megjelenítésének késését minimalizálni kell.
- A projekt hardver szinten bővíthető egy fejlettebb vezérlő egységgel, például Raspberry Pi felhasználásával, amely egy elválasztó réteget képezne a robot és a mobilalkalmazás között. A megközelítésnek megfelelően egyrészt nem a robot végezne bizonyos műveleteket, hanem ez az eszköz, másrészt a Raspberry Pi esetében alapértelmezetten csatlakoztatható egy kamera az eszközhöz, így késés nélkül feldolgozhatja a kapott adatokat.
- A projekten belül használt robotokkal kapcsolatban több hardver hiba is fellépett. Felmerült a terve egy saját robot megépítésének, amelynek vezérlésért, az előzőekben említett megközelítésnek megfelelően, egy Raspberry Pi lenne felelős.

Hivatkozások

1. ***, Az Atmel hivatalos oldala, Atmega8-as mikrokontroller leírása,
   http://www.atmel.com/devices/atmega8.aspx,
   utolsó megtekintés dátuma 2014.04.24
2. ***, Az Atmel hivatalos oldala, Atmega328p mikrokontroller leírása,
   http://www.atmel.com/devices/atmega328p.aspx,
   utolsó megtekintés dátuma 2014.04.24
3. ***, Assembly and Operational MANUAL Model ARX-03, Arexx Engineering, 2011.
4. ***, Arexx Arduino Robot Manual: AAR-04, Arexx Engineering, 2012.