

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1503

**OBRADA SONARSKOG SIGNALA ZA POTREBE
LOKALIZACIJE BESPILOTNE RONILICE**

Zdravko Eškinja

Zagreb, srpanj 2006.

Sadržaj:

1	UVOD	1
2	VIDEORAY PRO II	2
2.1	Tehničke karakteristike	2
2.2	Tritech Micron Sonar	2
2.2.1	CHIRP tehnologija	3
2.2.2	Širenje zvučnih impulsa	4
2.3	Kompas	5
2.4	Kontrolno sučelje	5
3	LABVIEW – kratki podsjetnik	6
4	LOKALIZACIJA BESPILOTNE RONILICE	7
4.1	Dvostrana komunikacija sonar – računalo	7
4.2	Projektiranje vlastitog software-a za komunikaciju	10
4.2.1	Kodiranje sonarske poruke	11
4.2.2	Dekodiranje sonarske poruke	20
4.2.3	mtAlive	26
4.2.4	mtSendData	26
4.2.5	Resetiranje sonara	27
4.3	Skeniran niz	27
4.4	Određivanje udaljenosti	28
4.5	Određivanje pozicije	28
4.5.1	Kružni bazen	29
4.5.2	Grafički prikaz rotiranja ronilice ovisno o smjeru orijentacije	36
4.5.3	Kvadratni bazen	38
4.6	Aplikacije za lokalizaciju	44
4.6.1	Čitanje i pisanje s porta	44
4.6.2	Spremanje podataka (logiranje)	45
4.6.3	Podešavanje brzine gibanja ronilice u svrhu lokalizacije ronilice	47
4.6.4	Aplikacija temeljena na skeniranju	48
4.6.5	Aplikacija temeljena na pozicioniranju	50

5	REZULTATI EKSPERIMENTALNIH ISPITIVANJA.....	53
5.1	Usporedba brzina različitih aplikacija	54
5.1.1	Aplikacija temeljena na pozicioniranju	54
5.1.2	Aplikacija temeljena na skeniranju	55
5.1.3	Zaključak usporednog testa.....	56
5.2	Eksperimentalno ispitivanje mogućnosti sonara.....	56
5.2.1	Utjecaj vremena diskretizacije skeniranog niza na brzinu lokalizacije	56
5.2.2	Utjecaj dometa (<i>range</i>) na brzinu	57
5.2.3	Utjecaj brzine komunikacije sonar - računalu na brzinu aplikacije.....	57
5.2.4	Utjecaj skeniranog niza na točnost lokalizacije	57
5.2.5	Izmjerena preciznost	64
5.2.6	Testiranje lokalizacije u stvarnom vremenu (<i>Online</i> Lokalizacija).....	67
6	ČESTO POSTAVLJANA PITANJA (FAQ).....	68
7	ZAKLJUČAK.....	69
8	LITERATURA	70
9	DODATAK A – algoritam za određivanje udaljenosti.....	71
10	DODATAK B – algoritam za određivanje pozicije.....	72
11	DODATAK C – algoritam za rotaciju i translaciju	74
12	DODATAK D – algoritam za pozicioniranje u pravokutniku.....	75

Popis oznaka i kratica

FER	Fakultet elektrotehnike i računarstva, Zagreb
NI	National Instruments
VI	Virtual Instruments
engl.	engleski
ROV	Remotely Operated Vehicles
AUV	Autonomous Underwater Vehicles
USB	Universal Serial Bus
I/O	input / output
CHIRP	Compressed High Intensity Radar Pulse
DST	Digital Sonar Technology
PC	Personal Computer

Popis tablica

Tablica 4.1 Značenje prvih 13 bajtova.....	11
Tablica 4.2 Objašnjenje mtHeadCommand poruke	12
Tablica 4.3 Objašnjenje parametra HdCtrl	14
Tablica 4.4 Primjer mtHeadData poruke	21
Tablica 4.5 Objašnjenje mtHeadData poruke	21
Tablica 4.6 Objašnjenje parametra 'HeadStatus'	22
Tablica 4.7 Objašnjenje parametra 'Sweep Code'	22
Tablica 4.8 mtAlive	26
Tablica 4.9 mtSendData.....	26
Tablica 4.10 mtReBoot	27
Tablica 4.11 Točke koje opisuju rubove prikaza ronilice.....	37
Tablica 4.12 Podaci koji se zapisuju u txt file	45
Tablica 4.13 Vrijedosti koeficijenata za skaliranje.....	48
Tablica 5.1 Rezultati prvog mjerenja.....	65
Tablica 5.2 Rezultati drugog mjerenja.....	66

Popis slika

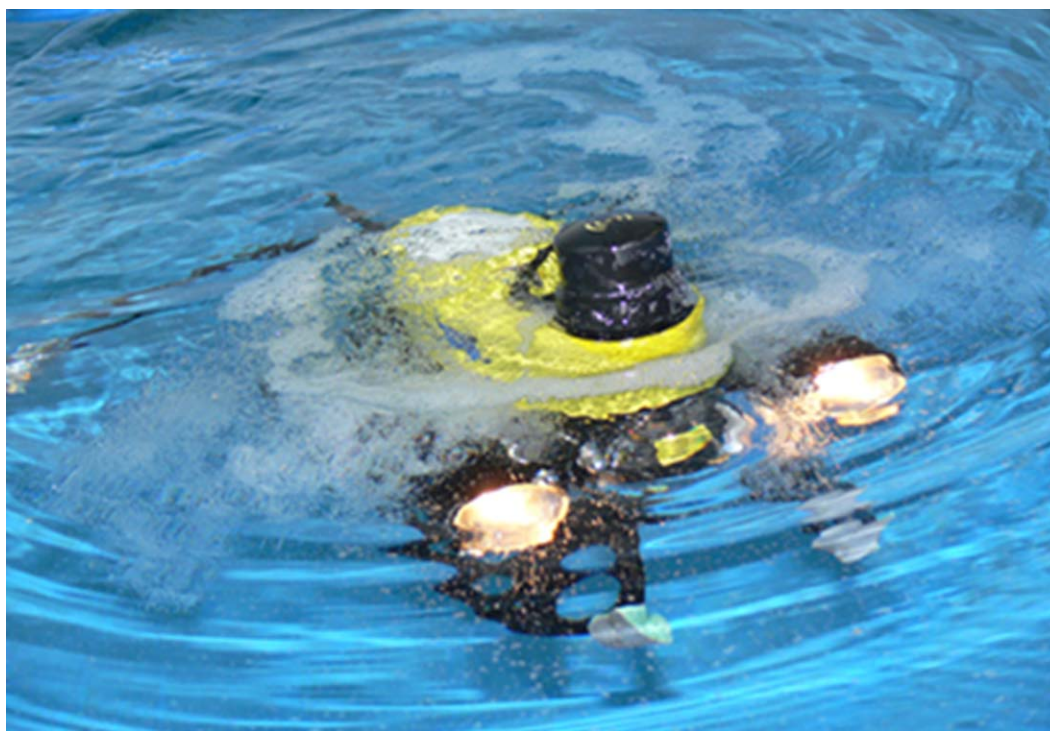
Slika 1.1 VideoRay Pro II.....	1
Slika 2.1 Tritech Micron sonar	3
Slika 2.2 Priključak za sonar.....	3
Slika 2.3 Impuls emitiranog signala.....	3
Slika 2.4 Širenje impulsa	4
Slika 2.5 Utjecaj vode i rubova na širenje impulsa.....	5
Slika 4.1 Shema komunikacije i konektori	7
Slika 4.2 Konektor za suho ispitivanje	8
Slika 4.3 Sučelje SeaNet Setup-a.....	9
Slika 4.4 Primjer sonarske slike.....	10
Slika 4.5 Dio blok dijagrama funkcije w2s.....	15
Slika 4.6 Blok dijagram funkcije dec2hex_split	16
Slika 4.7 Drugi dio blok dijagrama funkcije w2s	17
Slika 4.8 Prikaz skeniranog niz i značenje parametara ADLow i ADSpan.....	19
Slika 4.9 Korisničko sučelje funkcije w2s.....	20
Slika 4.10 Blok dijagram funkcije rfsSVI.vi	23
Slika 4.11 Blok dijagram funkcije rfsSVI.vi	25
Slika 4.12 Blok koji se nalazi pod <i>tru</i> opcijom.....	26
Slika 4.13 Skenirani niz	27
Slika 4.14 Određivanje udaljenost ronilice tj točke T u odnosu na točku S	29
Slika 4.15 Određivanje orijentacije udaljenosti x u odnosu na sjever tj. duljinu SN'	31
Slika 4.16 Prikaz problema simetričnosti	32
Slika 4.17 Računanje udaljenosti d2.....	33
Slika 4.18 Sučelje simulativne aplikacije	35
Slika 4.19 Blok dijagram funkcije koja crta ronilicu (kao točku).....	36
Slika 4.20 Blok dijagram funkcije koja crta ronilicu	38
Slika 4.21 Prikaz određivanja pozicije u kvadratnom bazenu	40
Slika 4.22 Praćenje pomaka kursa u odnosu na apcisu.....	42
Slika 4.23 Sučelje za kvadratni bazen.....	43

Slika 4.24 a) Blok dijagram za komunikaciju preko porta b) sučelje	44
Slika 4.25 Blok dijagram logiranja	46
Slika 4.26 Dio blok dijagrama PC-pilota gdje je	47
Slika 4.27 Blok Dijagram aplikacije koja se temelji na skeniranju	49
Slika 4.28 Blok dijagram aplikacije koja se temelji na pozicioniranju.....	51
Slika 5.1 Laboratorijski bazen	53
Slika 5.2 Neželjena refleksija od ruba	59
Slika 5.3 Neželjena refleksija od ronilice	59
Slika 5.4 Utjecaj neželjene refleksije od ruba.....	60
Slika 5.5 Rub ima maksimalnu amplitudu no početak njegova.....	60
Slika 5.6. Početno izdizanje jest odvojeno od maksimalnog izdizanja.....	61
Slika 5.7 Utjecaj plutače : plutača je 1m ispred, a očitana kao 1m iza ruba.....	61
Slika 5.8 Sonar je 20 cm ispod površie. Do izražaja dolazi kabel	61
Slika 5.9 Sonar se nalazi 10 cm ispod površine.....	62
Slika 5.10 Neobjašnjivo izdignuće na 1.6m.....	62
Slika 5.11 Refleksija od ruba kad je ronilica jako blizu ruba	63
Slika 5.12 Neželjena refleksija se nalazi od 0.6m.	63
Slika 5.13 Još jedan primjer neželjene refleksije od ronilice.....	64
Slika 5.14. Intezitet neželjene refleksije se smanjuje približavanjem rubu	64
Slika 5.15 Ronilica u trenutku mjerenja preciznosti	65

1 UVOD

Voda prekriva $\frac{3}{4}$ ukupne površine na planeti. A ispod te površine buja život, kriju se tajne i čuvaju bogatsva do kojih čovjek do sada još nije stupio. Čak 98 % morskog dna je ostalo neistraženo. Razvoj tehnologije i napredovanje svih grana znanosti omogućuje čovjeku da krene na put izvan svojih tjelesnih mogućnosti i proširi svoje dvorište u jedan potpuno novi svijet – morsku dubinu.

Cjevovodi, platforme, brane i mostovi tek su neki od suvremenih objekata koji zahtijevaju rad na dubini. Taj posao zasad obavljaju ronici, no ne još dugo. Autonomna podvodna vozila ključ su uspjeha na tom području: ljudska opasnost eliminirana, brzina rada povećana, cijena smanjena.



Slika 1.1 VideoRay Pro II

Poznati svjetski morski instituti i razna sveučilišta već se godinama bave razvojem algoritama koji će osamostaliti ronilicu i odvojiti je od direktne kontrole operatera. Jedan od najbitnijih problema je određivanje trenutne pozicije. Ovaj rad se ipak bavi puno jednostavnijim problemom, a to je lokalizacija u poznatoj okolini pravilnog oblika. Primjene su višestruke: kontrolno snimanje i ostale podvodne operacije u različitim strukturiranim

okruženjima: rezervoarima, tankovima, bazenima itd. Da bi to postalo uopće moguće potreban je rad u stvarnom vremenu što zahtjeva vrlo brze uređaje i još brže algoritme. Pri implementaciji je korištena ronilica VideoRay Pro II s Tritech Micron sonarom i programski paket LabVIEW 7.1. Razvijeni algoritmi i mogućnosti sonara su ispitani u laboratorijskom kružnom bazenu.

2 VIDEORAY PRO II

VideoRay Pro II je daljinski upravljano vozilo. Uz samu ronilicu uključena je i kontrolna ploča Integrated Control Box sa serijskim izlazom prema računalu.

2.1 Tehničke karakteristike

Maksimalna dubina	: 150 m
Brzina (Prema naprijed)	: 0-2 čvora
Dužina	: 355 mm
Širina:	: 228 mm
Visina	: 215 mm
Težina	: 3.5 kg
Ulazni napon	: 100 – 240 V AC, 50 – 60 Hz
Potrebna ulazna snaga	: oko 300 W
Temperaturni raspon	: 0 – 40 °C
Propulzori	: lijevi, desni i vertikalni u središnjem djelu ROV – a
Standardno osvjetljenje	: 2 halogena reflektora od po 20 W
Kabel	: 75 metarski, neutralnog uzgona
Prednja kamera	: fokus velikog raspona, nagib 160 stupnjeva, u boji
Stražnja kamera	: crno bijela

2.2 Tritech Micron Sonar

Tritech Micron sonar jest najmanji sonar na svijetu koji koristi CHIRP¹ signal i pripada grupi suvremenih DST² sonara.

¹ engl. Compressed High Intensity Radar Pulse

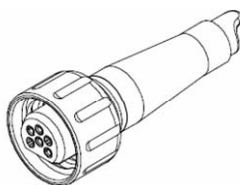


Slika 2.1 Tritech Micron sonar

Sastoji se od dva osnovna dijela:

- Sonarske glave koja se vrti unutar kućišta. Ona ima na sebi odašiljač zvučnog signal i odgovarajući prijemnik. Vrti se uz pomoć koračnog motora koji ima maksimalno 6400 pozicija.
- Vodonepropusno kućište koje ima priključak za dva kanala. Drugi kanal se koristi u interakciji s drugim uređajima.

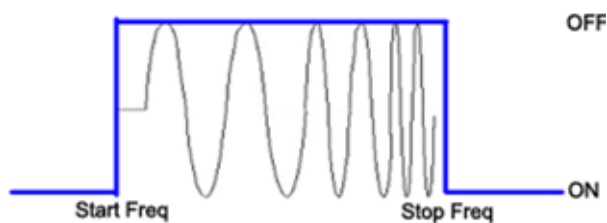
Na slici 1.1 je prikazan originalni priključak.



Slika 2.2 Priključak za sonar

2.2.1 CHIRP tehnologija

Impulsni signal koji se emitira prikazan je na slici 2.3. Duljina tog impulsa je vrlo važna jer određuje domet.



Slika 2.3 Impuls emitiranog signala

² engl. Digital Sonar Technology

Kao što je vidljivo iz slike ne koristimo samo jednu frekvenciju, već je frekvencija rasprostranjena unutar nekog opsega . U slučaju Micron sonara taj opseg iznosi od 650kHz do 750 kHz.. Na taj način impuls je označen ovisno o frekvenciji, pa sad sonar zna što je poslao i kada. Tehnikom prepoznavanja uzoraka određuje se koji dio impulsa dolazi prije.

Zahvaljujući tome sposobnost razlučivanja mete nije određen duljinom impulsa kao što je to slučaj s monokromatskim signalom: Rezolucija je dana relacijom (2-1).

rezolucija dometa = (2 x brzina zvuka) / raspon frekvencija

2-1

raspon frekvencija = Stop Freq–Star Freq

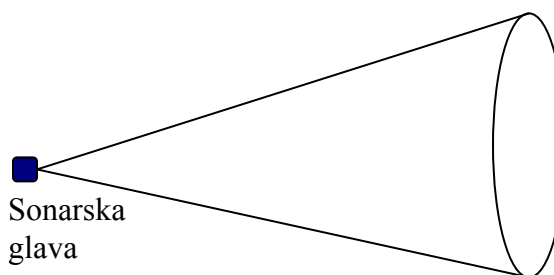
Najbolja moguća rezolucija jest 7.5 mm, i puno manje ovisi o dometu.

Ostale karakteristike:

- poboljšano odbacivanje šuma
- duži puls, a to znači veći domet
- smanjena potrošnja energije zvučnog signala.

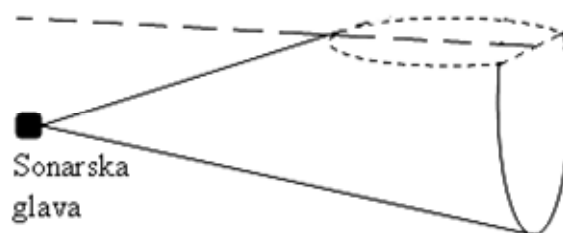
2.2.2 Širenje zvučnih impulsa

Sonar odašilje signal koji se širi u obliku stošca (vidi sliku 2.4), a prima dijelove signala koji se odbijaju. Veća amplituda odbijenog signala svjedoči o prepreci šireg opsega.



Slika 2.4 Širenje impulsa

U slučaju da ronilica nije dovoljno duboko, stožac presijeca razina vode jer je brzina širenja zvuka kroz zrak neusporedivo manja pa ne očitavamo ništa (vidi sliku 2.5).



Slika 2.5 Utjecaj vode i rubova na širenje impulsa

Suprotan učinak dobijemo ako je situacija kad zraku usmjerimo u udubljenje ruba bazena (otvori pumpe). Signal se i dalje širi, pa tu imamo pojavu dvostrukog ruba.

2.3 Kompas

Kurs određujemo pomoću digitalnog kompasa koji je implementiran u unutrašnjost ronilice. Rezolucija vrijednosti kursa iznosi 2° .

2.4 Kontrolno sučelje

Upravljanje ronilicom se obavlja preko originalnog korisničkog sučelja *PC Pilot* dizajniranog u programskom paketu LabVIEW.

3 LABVIEW – kratki podsjetnik

Ovo poglavlje je kratki podsjetnik ili najosnovniji opis velikog programskog paketa za one koji nisu odviše upoznati. Dio se orijentira na pojmove i metode korištene u razvijenoj aplikaciji.

U LabVIEW-u programiranje je strukturirano kao povezivanje blokova. Za tipične i standardne operacije postoje gotovi blokovi. Blokovi se izvršavaju slijedno, čekajući vrijednosti na ulazu. Završetkom operacije zadnjeg bloka završava rad VI³-aplikacije. To je bitno znati u radu s petljama, podfunkcijama, SubVI⁴-ima. Petlja treba završiti da bi sljedeći blok uzeo potrebnu vrijednost varijable iz petlje.

SubVI se radi tako da u prednjem panelu stisne desna tipka miša na malu ikonu samog SubVI-a (nalazi se gore desno) i uključi opcija *show connectors*. Klikom miša na željeni konektor i objekt na sučelju kreiraju se ulazi i izlazi SubVI-a .

Sub-VI se ugrađuje tako da u sučelju VI-a klikne desna tipka miša, odabere opcija *set control* i pronađe stvoreni SubVI.

Došao sam do spoznaje da LabView program ima nekoliko ozbiljnih nedostataka. Pri računanju kuta kosinusovim poučkom rezultat se zapisuje u svoj format pa LabVIEW čita krivi broj. Nužno je napraviti dodatnu konverziju formata. Nije moguća istodobna komunikacija preko više portova s različitim brzinama prijenosa u istoj aplikacij. Tehnička podrška se ne javlja već mjesecima unatoč mom višestrukom slanju.

Pri radu s globalnim varijablama se preporuča upisivanje vrijednosti iz samo jedne funkcije, dok ostale čitaju, da ne bi došlo do istodobnog pristupa ili nesinkroniziranog čitanja i pisanja.

³ Virtual Instrument – aplikacija tipična za LabVIEW s ekstenzijom *vi*

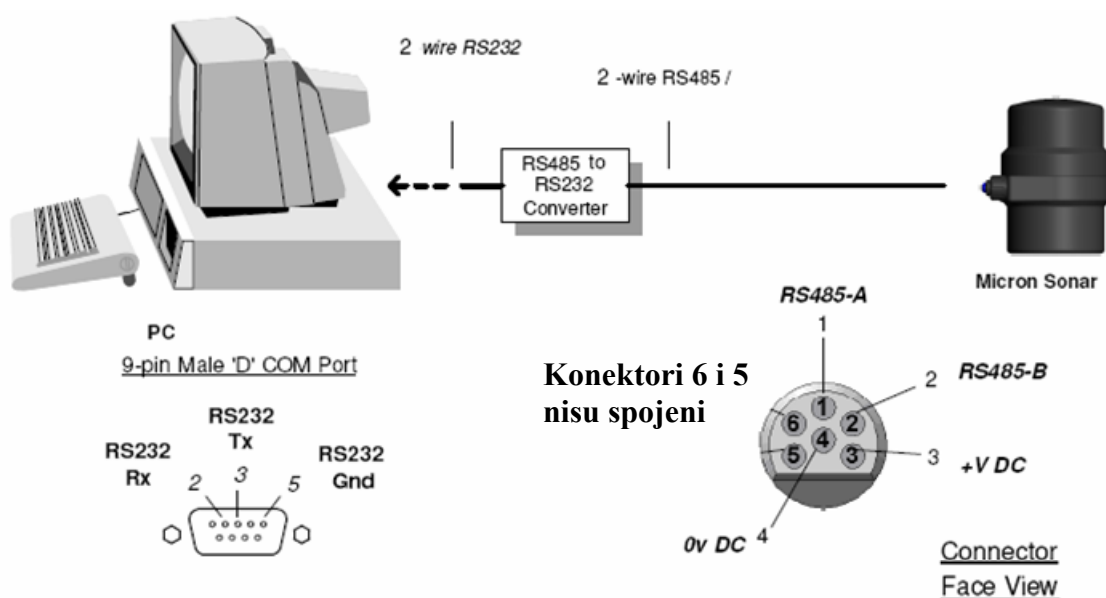
⁴ Aplikacija s određenim ulazima i izlazima koja se koristi kao funkcija u drugim aplikacijama

4 LOKALIZACIJA BESPILOTNE RONILICE

4.1 Dvostrana komunikacija sonar – računalo

Komunikacija sonara i računala odvija se preko RS-232 porta s tim da je potrebno izvršiti konverziju jer izvorno sonar ima samo četiri pina i „razumije“ samo RS-485 protokol.(vidi sliku 4.1).

Konverzija je izvršena pomoću uređaja *Optically isolated converter - model 4850 TLED tvrtke B&B electronics*. Ako je sonar spojen na ronilicu tada podaci dolaze iz kontrolnog panela s posebnim RS-485 kabelom, a napajanje je integrirano u kontrolnom pultu.



Slika 4.1 Shema komunikacije i konektori

Originalno se sve odvija preko komandnog pulta VideoRay Pro II Integrated Control Box.

U slučaju da se sonar želi ispitivati i koristiti neovisno o ronilici potrebno je dovesti istosmjerno napajanje od 12 V, a podatkovni pinovi moraju biti ukriženi: RS485-A od konvertera mora ići na RS485-B sonara i obrnuto.

Ovaj način suhog ispitivanja korišten je u početnoj fazi projekta, pri razvijanju aplikacija za kodiranje i dekodiranje. Korišten je konektor kojeg je osmislio i izradio Đula Nađ (vidi sliku 4.2).



Slika 4.2 Konektor za suho ispitivanje

Uz sonar postoji i SeaSprite software⁵ koji omogućuje vizualizaciju sonarske slike u različitim skenirajućim opcijama rada sonara. Spomenuti software oblikuje poruke koje se šalju u sonaru, no u sonaru također postoji software koji obrađuje te poruke i šalje odgovore. Podešavanje software-a u Micron sonaru vrši se preko SeaNet Pro programa. Korištene su dvije verzije programa: stariju (originalno kupljenu s sonarom) 1.12 i noviju 1.15.

Važno je napomenuti da se jedino sa novom verzijom može promijeniti čvor ('Node') koji predstavlja komunikacijsku adresu sonara, no pri svakoj promjeni dolazi do resetiranja sonara na RS232 komunikaciju. Tu se javlja problem jer izlaz iz sonara nije namijenjen RS232 komunikaciji. Potrebno je prespojavanje kako bi se u SeaNet setup-u vratile postavke za RS485 komunikaciju: Ovaj RS-232 protokol komunicira preko 3 žice s sonarom:

Pin1 → RS232 Tx, Pin2 → RS232 Rx, Pin5 → RS232 Gnd i to treba spojiti direktno (vidi sliku 4.1).

Svaki put kad se mijenja 'firmware' mora se prespojiti, a u Seanet setup-u 1.15 dolazi do sljedeće greške u slučaju korištenja konvertera sa RS 232 na USB priključak:

Error reading seRxFiltAlpha1.Value: '1000000000000' is not a valid integer value.

Iz tog razloga korišten je stariji software Seanet setup 1.12.

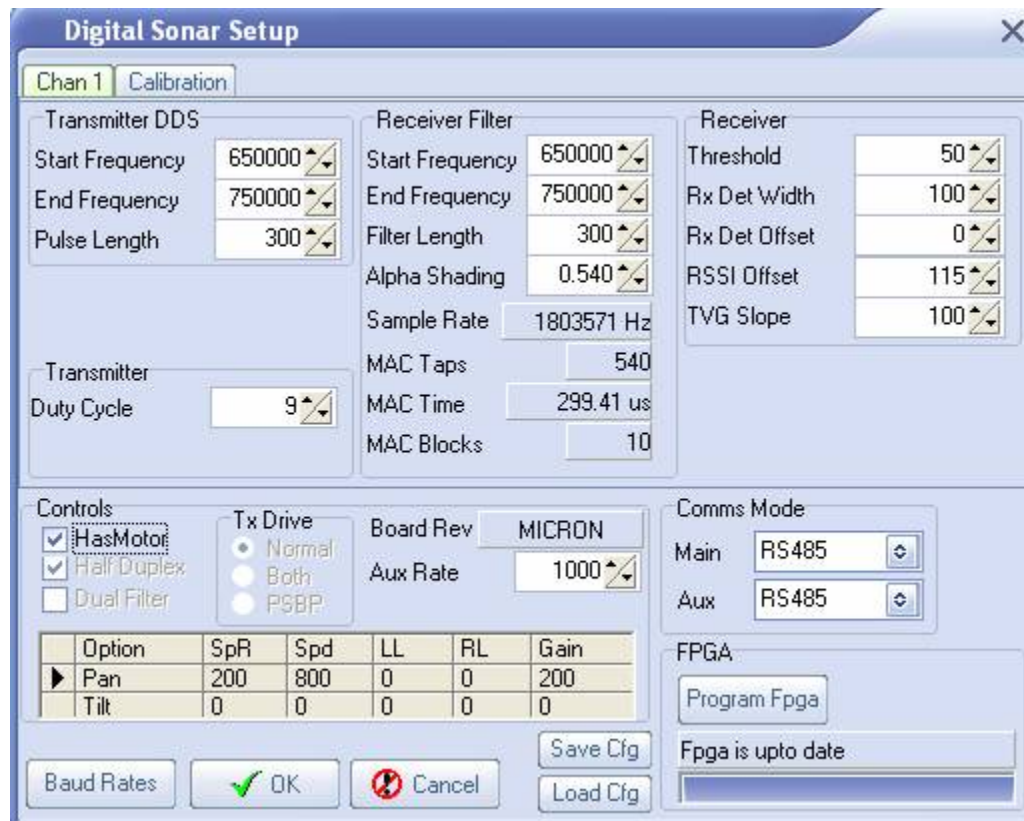
Ovaj Micron sonar ne može mijenjati frekvencije iako je ponuđena opcija već radi sa zvučnim signalom centralne frekvencije 700 kHz

Ako se V3B⁶ opcija ne koristi onda nisu potrebni parametri pa je poruka kraća, komunikacija brža.

⁵ programska podrška

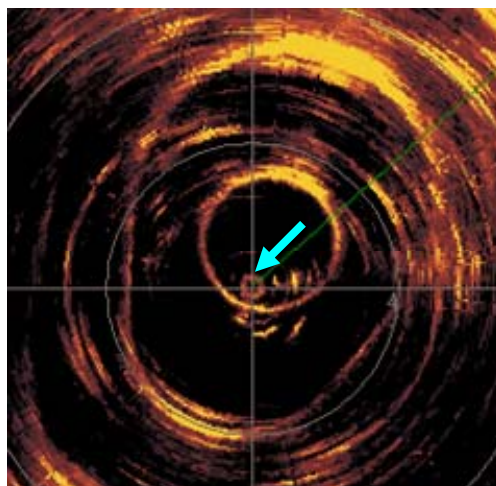
⁶ opcija za dijeljenje i ulančavanje velike poruke

Sve upute navedene u ovom dokumentu odnosit će se na Micron sonar koji ima RS-485 protokol ugrađen u 'glavi', a spojeno je na RS232 port računala preko optičkog konvertera. Na slici 4.4 se nalazi sučelje setup programa i opcije koje je moguće mijenjati.



Slika 4.3 Sučelje SeaNet Setup-a

Pri mijenjanju parametara u SeaNet Setup-u može se dogoditi da zbog prekratkog držanja tipke OK ili *Program FPGA*, software ne učitava sve podatke pa dolazi do greške *'Failed to program BBUser'*. Da bi radio njihov software SeaSprit Tx node (adresa sonara) mora biti 02. Sonarska slika u opciji rada gdje sonar kontinuirano skenira izgleda kao na slici 4.3.



Slika 4.4 Primjer sonarske slike

Kružić unutar kojeg se nalazi ronilica označen je sa strelicom, a predstavlja šum koju stvara sama ronilica. Oko kružića vrlo lako se može uočiti kružnica koja predstavlja rubove bazena, a svijetli obrisi unutar bazena su smetnje (stepenice, kabeli, plutača s klorom).

Svijetli obrisi koji okružuju rub bazena su rezultati refleksije sonarskog signala: naime zvuk se odbije od zida ali ne nestane već i dalje nastavi putovati. Kad se odbije od suprotnog ruba onda ga ponovno osjeti prijamnik i zapiše ga kao da se nalazi prepreka. To se pojavljuje iz razloga što sonar preračunava udaljenosti na temelju vremena potrebnog signalu da dođe do ruba bazena i vrati se. Taj efekt se eliminira ograničavanjem maksimalnog dometa.

4.2 Projektiranje vlastitog software-a za komunikaciju

Već postojeći software ne zadovoljava moje potrebe u ispitivanju svih mogućnosti rada sonara i primjeni sonarske slike u lokalizaciji. Zbog toga je bilo nužno napraviti vlastiti. Softver instaliran u sonarsku glavu razumije više od 25 različitih naredbi. Za normalni način rada koriste se samo četiri i koristimo ih na slijedeći način:

Za očitavanje jedne udaljenosti potrebno je sonaru poslati dvije poruke

- *mtHeadCommand* – parametri potrebni za rad (domet, pozicija, orijentacija...)
- *mtSendData* – naredba za dohvat željene informacije o udaljenosti.

Sonar na svaku od poruka odgovara s:

- *mtAlive* – potvrda o radu sonara (šalje se u periodu od 1 Hz)

- *mtHeadData* – informacija o udaljenosti.

Naredba *mtDataSend* se tokom rada ne mijenja, a naredbu *mtAlive* nije potrebno čitati.

Svaki poslani parametar mora biti kodiran u sonarski kod. To iziskuje zasebne funkcije za kodiranje naredbe *mtHeadCommand* i dekodiranje *mtHeadData*.

Prvih 13 bajtova kod svih poruka imaju isto značenje i ono je prikazano tablicom:

Tablica 4.1 Značenje prvih 13 bajtova

Bajt 1	Zaglavlje poruke = @
Bajt 2 - 5	Heksadekadska duljina binarnog paketa počevši od 6. bajta
Bajt 6,7	Binarna riječ od gore navedene udaljenost
Bajt 8	Identifikacijski broj dolaznog paketa (Tx Node broj 0 - 240)
Bajt 9	Identifikacijski broj odlaznog paketa (Rx Node broj 0 – 240)
Bajt 10	Broj bajtova poruke od 11. bajta
Bajt 11	Vrsta poruke – identifikacijski broj
Bajt 12	'Sequence Bitset' poruke
Bajt 13	Identifikacijski broj dolaznog paketa (Tx Node broj 0 - 240)

4.2.1 Kodiranje sonarske poruke

Poruka *mtHeadCommand* (ID 19⁷) sadrži parametre uređaja (zvane '*Device parameters*') od 15. do uključivo 68. bajta.

Primjer takve poruke dan je u tablici 4.2.

⁷ ID je identifikacijski broj naredbe a zapisan je u 11 bajtu

Table 4.1 mtHeadCommand poruka

40	30	30	34	43	4C	00	FF	02	47	13	80	02	1D
Hdr '@'	Hex Length = 76 bytes				Bin Length = 76 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 71	Mt H'd Cm d	Seq = End	Nde 02	V3B Para -ms
83	23	02	99	99	99	02	66	66	66	05	A3	70	3D
HdCtrl * = 9091		HdT ype = 02	TXN, Ch1 = 43620761				TXN, Ch2 = 90596966				RXN, Ch1 = 104689827		
06	70	3D	0A	09	28	00	3C	00	01	00	FF	18	51
	RXN, Ch2 = 151666032				TxPulseLen = 40 usec		RangeScale = 6 metres		LeftLim = 1 (1/16 Grad)		RightLim = 6399 (1/16 Grad)		AD Sp'n = 81
08	54	54	5A	00	7D	00	19	10	8D	00	5A	00	E8
AD Low = 8	Iga- in, Ch1	Iga- in, Ch2	Slope, Ch1 = 90		Slope, Ch2 = 125		Mo' Tme = 25	Step Size = 16	ADInterval = 141		Nbins = 90		Max ADb uf
03	97	03	40	06	01	00	00	00	50	51	09	08	54
= 100 0	Lockout = 919 usec		MinorAxis Dir = 1600 (1/16 Grad)		Maj' Axis Pan	Ctl2 = 0	ScanZ = 0		AD Sp'n Ch1	AD Sp'n Ch2	AD Low Ch1	AD Low Ch2	Iga- in Ch1
54	00	00	5A	00	7D	00	00	00	00	00	0A	LF	
Iga- in, Ch2	Adc SetP Ch1	Adc SetP Ch2	Slope, Ch1 = 90		Slope, Ch2 = 125		Slope Delay, Ch1 = 0		Slope Delay, Ch2 = 0				

Objašnjenje svakog bajta zasebno je prikazano u tablici 4.3.

Tablica 4.2 Objašnjenje mtHeadCommand poruke

Bajt 14	Tip mtHeadCommand poruke: "1" = Normal "29" = s dodanim V3B načinom rada parametrima za opciju rada s dvostrukim kanalom
Bajtovi 15-16	'Hdctrl' ⁸ - bajtovi o kojima ovisi načini rada sonarske glave
Bajt 17	'HdType' - Tip uređaja (za sonar 02)
Bajtovi 18-25	'Txn' - Broj odašiljača za oba kanala
Bajtovi 26-33	'Rxn' - Broj prijemnika za oba kanala
Bajtovi 34-35	'TxPulseLen' – Duljina impulsa odašiljača (u mikro sekundama)
Bajtovi 36-37	'RangeScale' – domet u decimetrima
Bajtovi 38-39	'Llim' - Lijeva kutna granica pri opije skeniranja sektora (1/16 Gradijana)

⁸ Naredba 'Hdctrl' bit će objašnjena u nastavku

Bajtovi 40-41	'Rlim' - Desna kutna granica pri opije skeniranja sektora (1/16 Gradijana)
Bajt 42	'AdSpan' ⁹ širina 'prozora ' (1/16)
Bajt 43	'AdLow' minimalna amplituda (1/16) - vrijedi samo za 4 bitno uzorkovanje
Bajtovi 44-45	'Igain' – početno postavljena vrijednost pojačanja amplitude za oba kanala (1/210)
Bajtovi 46-49	'Slope' – regulacija vremenski promjenljivog pojačanja
Bajt 50	'MoTime' – Vremenska konstanta koraćnog motora (ne utječe)
Bajt 51	'Step' – Korak motora u kutovima (1/16 Gradijana)
Bajtovi 52-53	'AD Interval' – period uzorkovanja 'ping-a' ¹⁰
Bajtovi 54-55	'Nbins' Broj 'binova' (uzoraka) u jednom skeniranom nizu
Bajtovi 56-57	'MaxADbuf' – MaxADbuf=500 ;Granica 1000
Bajtovi 58-59	'Lockout' – trenutak u mikro sekundama nakon odašiljanja signala kad treba početi slušati
Bajtovi 60-61	'Minor Axis' – opcija za dvoosne sonare – za jednoosne =1600
Bajt 62	'Major Axis' – opcija za dvoosne sonare - za jednoosne = 1
Bajt 63	'Ctl2' – dodatne kontrole u svrhu ispitivanja = 0
Bajtovi 64,65	'ScanZ' – opcija za specijalne uređaje = 0
Bajtovi 66-81	Služe za podešavanje parametara u slučaju ulančanih poruka ('V3B Gain Parameters')

Ne koristim ni *V3B* opciju ni ulančavanje poruke pa to neće biti detaljno objašnjeno.

Sve vrijednosti su zapisane heksadekadski, a u slučaju vrijednosti veće od 1 bajta onda se heksadekadski zapis bajta najniže vrijednosti piše prvi i tako redom, vidi tablicu 4.3 npr. 15 i 16 bajt.

⁹ AdSpan I Ad Low opcije vrijede samo za 4 bitno uzorkovanje iako u dokumentaciji piše da vrijedi za oba načina uzorkovanja. Kod 8 bitnog uzorkovanja ova opcija ne utječe.

¹⁰ kontinuirani oblik skeniranog niza tj. očitavanje sonara

U tablici 4.4 objašnjen je bit po bit naredbe 'Hdctrl' koja je najbitnija a za postavljanje parametara u sonarsku glavu.

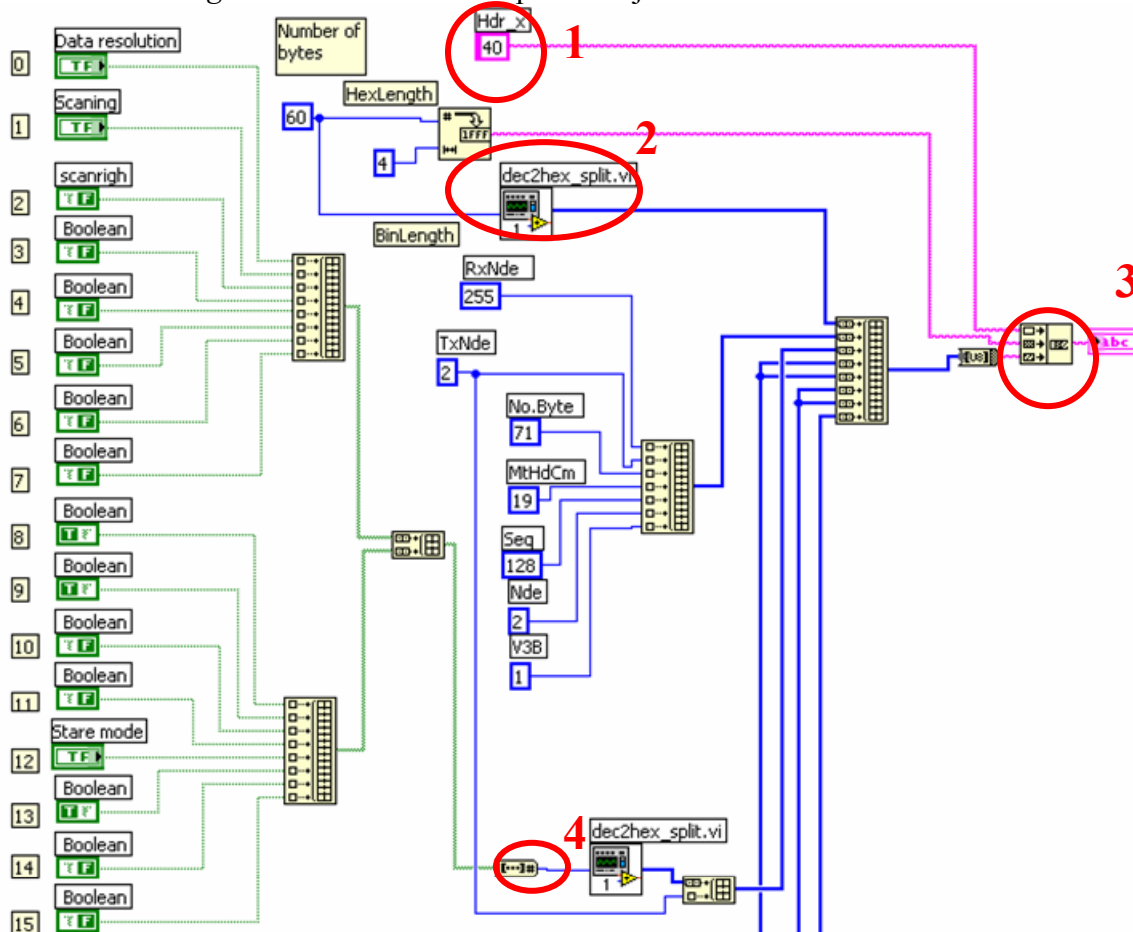
Tablica 4.3 Objašnjenje parametra HdCtrl

Bit 0	'adc8on'	Rezolucija skeniranog niza: 0=4bitna širina uzorka skeniranog niza tj Bin-a, 1=8 bit
Bit 1	'cont' ‘	Način skeniranja: 0=Djelomično Skeniranje, 1=Kontinuirani)
Bit 2	'scanright	Smjer Skeniranja: 0=Lijevo, 1=Desno
Bit 3	'invert'	Orijentacija Sonarske glave: 0= Naprijed, 1=Obrnuto– netočno – naprijed je uvijek
Bit 4	'motoff'	Rad motora: 0=Motor radi , 1=Motor ne radi - – ne utječe- uvijek radi
Bit 5	'txoff'	Onesposobljavanje odašiljača u svrhu ispitivanja 0=Tx uključen, 1=Tx isključen.
Bit 6	'spare'	Rezervirani bit: 0=Normalno / uvijek
Bit 7	'chan2'	Odabir Kanala: 0=Koristi Kanal 1, 1=Koristi Kanal 2 hSSS preusmjeren preko opcije ping pong
Bit 8	'raw'	Upravljački bit razine sustava – uvijek 0
Bit 9	'hasmot'	Postojanje motora: 0= Nema Motor , 1=Ima Motor
Bit 10	'applyoffset'	Dinamički pozicioniranje u odnosu na neku referencu, najčešć sonar: 0=Zanemari
Bit 11	'pingpong'	Opcija alterniranja pri korištenju dvo frekvencijskog kanala za sonare koji mjere strane: 0= za sonar koji skenira
Bit 12	'stareLLim'	Opcija : 1=ne skeniraj već pozicioniraj se na lijevu granicu
Bit 13	'ReplyASL'	Bti je postavljen u 1 tako da uzorci (binovi) iz skeniranog niza budu analogni
Bit 14	'ReplyThr'	Bit rezerviran za specijalne funkcije uvijek 0
Bit 15	'IgnoreSensor'	Bit za dijagnozu: 1=Zanemari grešku na sonaru

4.2.1.1 Realizacija postupka kodiranja u programskom paketu LabVIEW

Zbog neprestane promjene parametara i načina rada sonarske glave uslijed skeniranja, dizajnirana je aplikacija (w2s.vi - 'write to string') koja samostalno kodira parametre i stvara valjanu 'mtHeadCommand' poruku.

Jedan dio 'Block Diagram'-a SubVI-a 'w2s' prikazan je na slici 4.5.



Slika 4.5 Dio blok dijagrama funkcije w2s

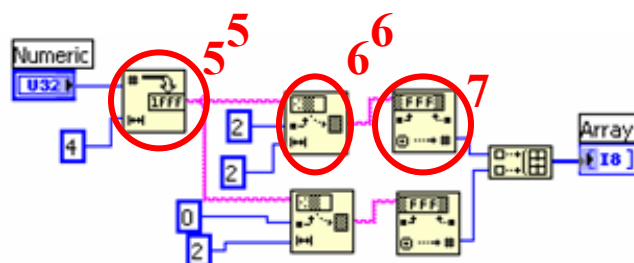
Svi parametri prije zapisa u poruku moraju biti kvalitetno obrađeni i zapisani u format string:

Prvi bajt poruke je zaglavlje i to znak @ zapisan u heksadekadskom obliku. (1)¹¹ Realiziran je kao 'konstanta string' (niz karaktera) od jednog znaka. Tu je to bilo moguće jer je zaglavlje nepromjenjivo, no većina ostalih parametara su nepromjenjivi i potrebne su različite modifikacije brojeva. Nakon modifikacija dobiju se stringovi koje se spoje po točno

¹¹ Radi lakšeg snalaženja bitne djelove sa ove sheme sam označio i pridodao odgovarajući broj

određenom redosljedu pomoću već postojećeg bloka *'Concatenate string'* (3), i novo dobiveni string je željena kodirana poruka.

U sljedeća četiri bajta je zapisana duljina binarnog paketa u heksadekadskom obliku koja zajedno sa binarnim zapisom (sljedeća 2 bajta) čine sustav provjere točnosti prijenosa – jednostavno se usporede vrijednosti ta dva broja. U aplikacijama nije implementiran algoritam usporedbe. Prvi slučaj zahtjeva pretvorbu udaljenosti *'Number of bytes '* (za naš mtHeadCommand to je broj 60) u heksadecimalni string što znači da svaku znamenku broja *'Number of bytes'* =0060 pretvara u ASCII zapis i najniže bajtove jedan za drugim (string format). U drugom slučaju taj isti broj pretvaramo u čisti heksadekadski format, no prije zapisa potrebno je obrnuti redosljed bajtova. Kako je ova pretvorba česta i koristim je gotovo za sve parametre napravio sam vlastiti blok *'dec2hex_split.vi'* (2) čija je shema prikazana na slici 4.6.



Slika 4.6 Blok dijagram funkcije dec2hex_split

Blok služi za modificiranje brojeva u 2-bajtni zapis. Broj koji je potrebno modificirati prvo pretvaramo u heksadekaski string (5), zatim ga raščlanjujemo na dva stringa veličine 1 bajt (6). Novodobivene heksadekadske stringove pretvaramo u brojeve(7) i njih upisujemo u polje bajtova (*Build Array.vi*) obrnutim redosljedom. Broj koji predstavlja bajt niže vrijednosti (prije dobivenog heksadecimalnog zapisa tj. stringa)zapisujemo kao prvog i tako redom. Izlazna varijabla ovog bloka je polje bajtova pa prije ulančavanja u string potrebna je konverzija *'Byte Array To String'* (slika do bloka 3). Ovim blokom omogućujemo upis decimalnog zapisa.

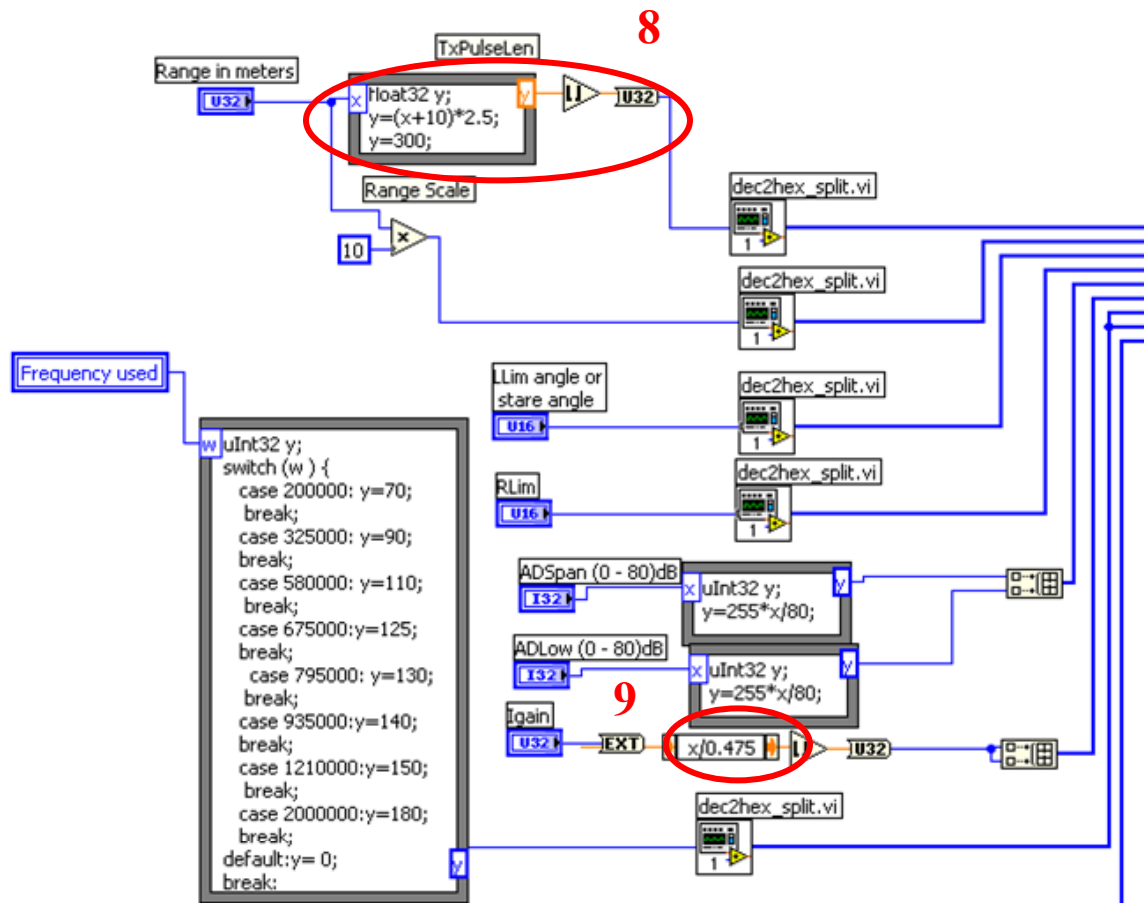
Funkcija za 4 bajtni zapis se razlikuje u tome što heksadekadski string ne raščlanjujemo na dva nego na četiri stringa. Na slici 4.5 vidimo da izlaz iz SubVI-a završava u

bloku za stvaranje polja ('*Build Array*'). Ovaj blok se razlikuje od onog prije navedenog jer ima uključenu opciju 'concatenate string'¹².

Parametri koji se nikada ne mijenjaju tokom ispitivanja, a veličine su jednog bajta npr TxNde i RxNde direktno su zapisani kao heksadekadski brojevi i spojeni u polje.

Brojevima od 0-15 na lijevoj strani slike označeni su bitovi 'HdCtr' parametara. Kao što se vidi iz slike realizirani su logičkim ('*boolean*') blokovima pri čemu bitovi '0' '1' i '12' nisu konstante već kontrole koje imaju pripadajući objekt na sučelju. Nakon stapanju u polja od jednog bajta slijedi pretvara polja logičkih varijabli u broj (4).

Na slici 4.7 je nastavak sheme w2s.vi.



Slika 4.7 Drugi dio blok dijagrama funkcije w2s

¹² Opcija *Concatenate* ulančava različita u jedno cjelovito polje. Bez ove opcije dobiva se dvodimenzionalno polje.

Parametar *Range in Meters* je potrebno još pretvoriti i u decimetre. Ovaj parametar kao i *Frequency used* koristi pri proračunu drugih parametara. Stoga one služe kao ulazne varijable u blok 'Formula Node' gdje se odvijaju matematički proračuni ili programska selektiranje. Neki drugi parametri npr. 'Igain' koristi blokove za jednostavnije proračune:

- 'Expression Node' – matematički proračun s jednom varijablom
- 'To Extended Precision Float' - pretvorba u format Extended Precision Float
- 'Round To – Infinity' – zaokruživanje na minimum
- 'To Unsigned Long Integer' – pretvorba u format Unsigned Long Integer'.

Specifičnost varijable 'Formula Node' (8) jest da je lokalna. Lokalne varijable su zapisane na nekoj memorijskoj lokaciji a koriste se gdje bi provlačenje poveznih linija uzrokovalo nepreglednost ili tamo gdje je potrebno trenutno očitavanje vrijednosti u nekim blokovima npr. petljama.

Pretvorbe i vrijednosti najbitnijih parametara:

Txn & Rxn (Bajtovi 18-33)- Konstante odašiljača

$$TxN^{13} = F \cdot 2^{32} / 32e6$$

$F =$ 'Frequency used' – frekvencija odašiljača [Hz]

$$RxN = (F + 455000) \cdot 2^{32} / 32e6$$

Domet (Bajtovi 36,37) – izraženi u decimetrima

Lijevo i desno granični kutovi se zapisuje kao pozicija motora, a minimalni korak motora je 1/6 gradiana odnosno 0.056° (1 gradian = 1.8°).

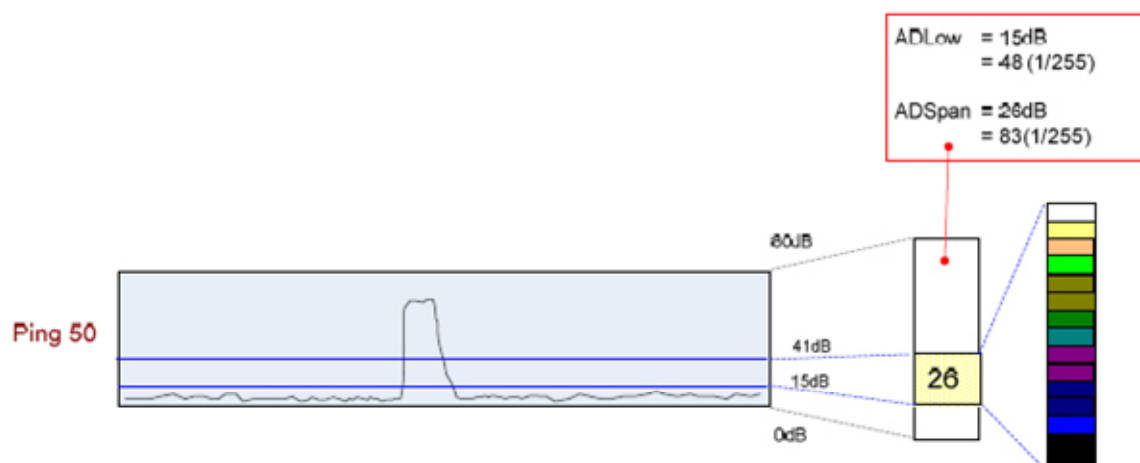
Neke vrijednosti:

Lijevo 90° -	1600
Naprijed -	3200
Desno 90° -	4800
Nasuprot -	0 (ili 6399)

¹³ Micron sonar u našem slučaju koristi samo jedan kanal za slanje i primanje signala pa nije moguće vršiti ove dvije operacije na različitim frekvencijama. Stoga se preporuča da bude chan1=chan2

AD Span (Bajt 42)/ AD Low (Bajt 43)

Njihovo značenje najbolje objašnjava slika 4.8.



Slika 4.8 Prikaz skeniranog niz i značenje parametara ADLow i ADSpan

Dane vrijednosti su iz jednog praktičnog primjera. Ping jest kontinuirani oblik sonarskog skeniranog signala iz kojeg se uzorkovanjem dobije skenirani niz. S parametrom *ADLow* definiramo minimalne vrijednosti amplitude koje uzimamo u obzir. *ADSpan* je širina prozora koje obuhvaćamo počevši od vrijednosti *ASLow*.

Ovisno o tome je li uzorak veličine 4 ili 8 bita, vrijednosti uzorkovanih amplitude se eskaliraju u intervale 0-16 i 0-255. Radi lakšeg očitavanja u software-u SeaSprite sonar taj interval je transformiran u spektar boja – kao na slici 4.8.

Iako bi trebao, eksperimentalno je pokazano da korišteni sonar pri 8 bita ne reagira na postavke ovih parametara već skalira signal od najveće očitane amplitude do minimalne. Zbog takvih okolnosti taj se problem rješava podešavanjem opcije *Igain*¹⁴.

Važno je napomenuti da za potrebe lokalizacije ova opcija nije od velikog značaja jer se ne može znati u kojim se intervalu nalaze tražene amplitude, a udaljenosti i domet su promjenjivi.

Slope(46 – 49) - regulacija vremenski promjenjivog pojačanja

Neke od vrijednosti:

325 kHz - Preporučeno i tvornički postavljeno 90

675 kHz - Preporučeno i tvornički postavljeno 125

¹⁴ Objašnjeno u poglavlju 4.2.1 u tablici 4.3

935 kHz - Preporučeno i tvornički postavljeno 140

AD Interval

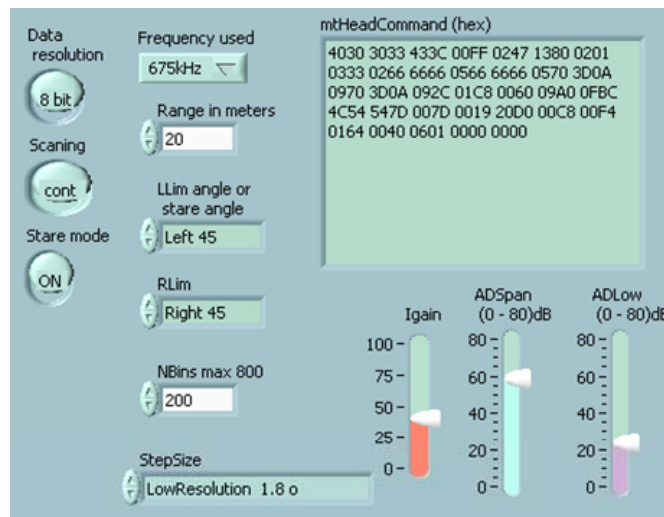
- ovisi o broju uzoraka skeniranog niza (broju binova) N_{Bins} -a

VOS (Brzina zvuka kroz vodu) = 1500 m/s

TT (Vrijem potrebno za uzorkovanje (dohvat) skeniranog niza (ping-a) = $2 * Domet / OS$

AD Interval = Cijeli broj veći i najbliži vrijednosti (TT/Nbins / 640 ns)

Iz korisničkog sučelja je moguće mijenjati 12 parametara, a sama aplikacija osim samostalnog rada izvodi se i kao objekt u složenijoj aplikaciji.



Slika 4.9 Korisničko sučelje funkcije w2s

4.2.2 Dekodiranje sonarske poruke

Poruka mtHeadData je odgovor sonara na njemu poslanu mtSendData poruku. Ona sadrži sve uzorke jednog skeniranog niza, i dio parametara važnih za rad sa sonarskom slikom. Dekodiranje je obrnuti postupak od sastavljanja poruke (kodiranja).

Primjer mtHeadData poruke prikazan je u tablici 4.1.

Tablica 4.4 Primjer mtHeadData poruke

40	30	30	35	34	54	00	02	FF	00	02	80	02	4C
Hdr '@'	Hex Length = 84 bytes				Bin Length = 84 bytes		Tx Nde 02	Rx Nde 255	Sin- gle pckt	mtH ead D'ta	Seq = End	Nde 02	->
00	02	10	05	85	A3	3C	00	66	66	66	05	6B	7D
Cou nt = 76	= Son head	Stat- us	Sw- EEP	HdCtrl = 41861		Range = 6m		TxN = 1717986821				Gain = 51%	->
00	32	2C	00	00	6B	00	40	06	C0	12	10	80	0A
Slo- pe = 125	AD Sp'n	AD Low	Heading Offset = 0		AD Interval = 107		L.Limit = 1600		R.Limit = 4800 (1/16 Grad)		Ste- ps = 16	Bearing = 2688 (1/16 Grad)	
2D	00	31	4B	78	76	75	65	4D	31	16	10	00	00
Dbytes = 45		Bin 1	Bin 2	Bin 3	Bin 4	Bin 5	Bin 6	Bin 7	Bin 8	Bin 9	Bin 10	Bin 11	Bin 12
00	00	00	00	00	00	00	00	00	00	00	00	00	00
Bin 13	Bin 14	Bin 15	Bin 16	Bin 17	Bin 18	Bin 19	Bin 20	Bin 21	Bin 22	Bin 23	Bin 24	Bin 25	Bin 26
00	00	00	00	00	00	00	00	00	00	00	00	00	00
Bin 27	Bin 28	Bin 29	Bin 30	Bin 31	Bin 32	Bin 33	Bin 34	Bin 35	Bin 36	Bin 37	Bin 38	Bin 39	Bin 40
00	00	00	00	00	0A								
Bin 41	Bin 42	Bin 43	Bin 44	Bin 45	LF								

Objašnjenje svakog bajta zasebno je prikazano u tablici 4.5.

Tablica 4.5 Objašnjenje mtHeadData poruke

Bajtovi 14,15	Ukupni broj bajtova koji slijede poslije ovog bajta (ne uključuje znak za kraj poruke (x0A)
Bajtovi 16	'HdType'- Tip uređaja (za 'Imaging hSON' sonar 02)
Bajt 17	'Head Status' bajt za provjeru sustavnog stanja sonarske glave
Bajtovi 18	'SweepCode' – Bitovi u kojima je zapisan način skeniranja
Bajtovi 19-20	'Hdctrl' - bajtovi o kojima ovisi načini rada sonarske glave
Bajtovi 21,20	'RangeScale' – domet u decimetrima
Bajtovi 23-26	'Txn' - Broj odašiljača – zanemari
Bajt 27	'Igain' – početno postavljena vrijednost pojačanja amplitude za oba kanala (1/210)
Bajtovi 28-29	'Slope' – regulacija vremenski promjenjivog pojačanja

Bajt 30	'AdSpan' širina 'prozora ' (1/16)
Bajt 31	'AdLow' minimalna amplituda (1/16) - vrijedi samo za 4 bitno uzorkovanje
Bajtovi 32,33	'Igain' – početno postavljena vrijednost pojačanja amplitude za oba kanala
Bajtovi 34,35	'Slope' – regulacija vremenski promjenjivog pojačanja
Bajtovi 36,37	'Llim' - Lijeva kutna granica pri opciji skeniranja sektora (1/16 Gradijana)
Bajtovi 38,39	'Rlim' - Desna kutna granica pri opciji skeniranja sektora (1/16 Gradijana)
Bajt 40	'Step' – Korak motora u kutovima (1/16 Gradijana)
Bajtovi 41,42	Trenutni položaj motora
Bajt 43	'Dbytes' broj uzoraka (binova) skeniranog niza koji slijede nakon ovog bajta

'Head Status' bajt opisuje stanje sonarske glave . Značenje svakog bita pojedinačno prikazano je u tablici 4.6.

Tablica 4.6 Objašnjenje parametra 'HeadStatus'

Bit 0	'HdPwrlLoss' Sonarska glava je u uvjetima resetiranja
Bit 1	Heksadekadska udaljenost binarnog paketa počevši od 6 bajta.
Bitovi 2,3	Binarna riječ od gore navedene udaljenost
Bit 4	Identifikacijski broj dolaznog paketa (Tx Node broj 0 - 240)
Bitovi 5,6	Identifikacijski broj odlaznog paketa (Rx Node broj 0 – 240)
Bit 7	Broj bajtova poruke od 11 bajta

Značenje pojedinačnih vrijednosti bajta 'Sweep Code' naveden je u tablici 4.7.

Tablica 4.7 Objašnjenje parametra 'Sweep Code'

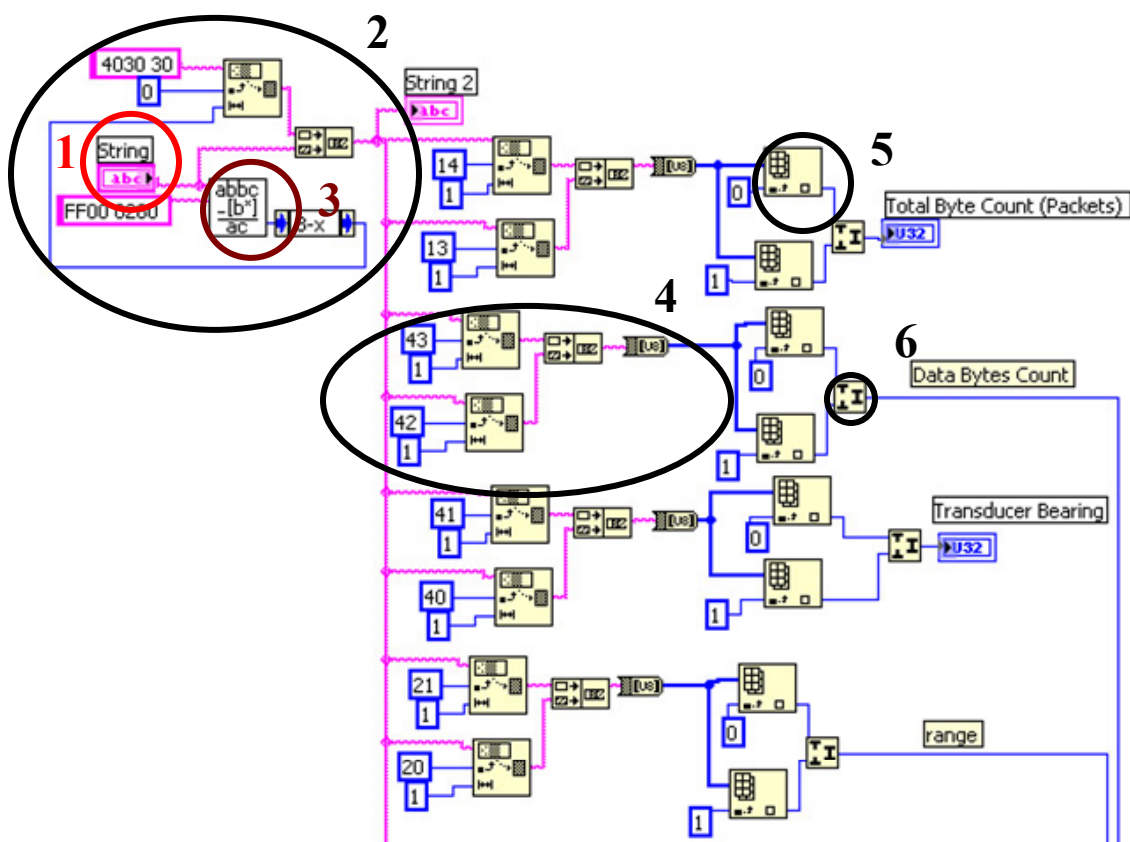
'0'	Normalno Skeniranje
'1'	Skenira na poziciji lijeve granice
'2'	Skenira na poziciji desne granice
'3','4'	Rezervirano – ZANEMARI
'5'	Skenira sa centralne pozicije (Naprijed – pozicija 0)

4.2.2.1 Realizacija dekodiranja u programskom paketu LabVIEW

Podfunkcija koja se dekodira *mtHeadData* poruku zove se **rfsSVI.vi** – 'read string Sub VI' i sastoji se od metoda koje izvlače skenirane uzorke i korisne parametre iz poruke. Ulazni podatak ove podfunkcije je razumljivo string varijabla (1). (Vidi sliku 4.10).

Prije postupka raščlanjivanja poruke bilo je potrebno izvesti funkciju¹⁵ (2) za neosjetljivost gubitaka prvih nekoliko bajtova.

Pri čitanju poruke¹⁶ događa se da se pri komunikaciju izgubi čak do 3 bajta i to u vremenski promjenjivim intervalima osobito pri korištenju RS232 to USB konvertera i ostalih elemenata koji unose kašnjenje. Taj gubitak informacije se može sanirati jer su prvih nekoliko bajtova uvijek nepromjenjivi ili služe za kontrolu prijenosa, pa su njihove vrijednosti¹⁷ poznate.



Slika 4.10 Blok dijagram funkcije rfsSVI.vi

¹⁵ Pod funkcijom u ovom slučaju se smatra određeni skup blokova namijenjeni obavljanju određene operacije

¹⁶ Realizacija čitanja i pisanja s porta bit će objašnjena u sljedećem poglavlju

¹⁷ Objašnjenje svakog bajta *mtHeadData* poruke je navedeno u prethodnom potpoglavlju

Vrijednosti 8., 9., 10., i 11. bajta su za ovu poruku i korišteni sonar nepromjenjivi, stoga vrijednost xFF00 0280 tražim u pročitanoj stringu *String* s blokom *Search and Replace Pattern.vi* (3) koji vraća indeks znaka u stringu na kojem se nalazi prvi znak pronađenog niza znakova. Tako znamo koliko je bajtova izgubljeno, pa tu vrijednost dovučem u blok za raščlanjivanje stringova (*String Subset.vi*), kreiram izgubljeni string i jednostavno ga dodam pročitanoj stringu.

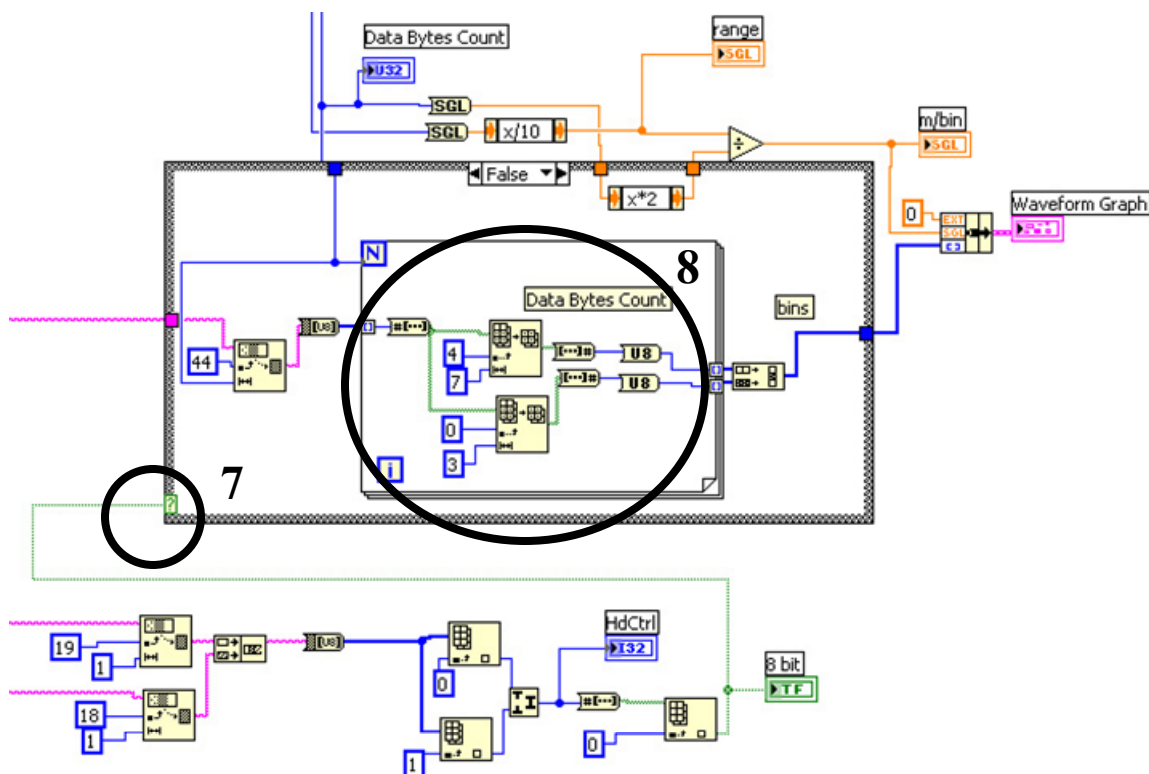
Slijedi skup blokova za raščlanjivanje. Prvo se kodirani parametar zapisan u stringu raščlani u stringove po jedan bajt, obrne se raspored znakova i spoji u zasebni string. Novodobiveni string se pretvori u polje bajtova. Iz polja se blokom '*Index Array*'¹⁸ (5) razluče elementi. '*Join Numbers.vi*' (6) spaja elemente u odgovarajući broj uzimajući u obzir da je prvi element skup bitova veće vrijednosti.

Kao što se da zaključiti ovaj postupak je inverzija kodiranja za svaki potreban parametar, a potrebni parametri su:

- Total Byte Count (Packets)
- Data Bytes Count
- Transducer Bearing
- Range
- HdCtrl – bit koji govori o rezoluciji bina (skeniranih uzoraka)
- m/bin- pomak u metrima po binu .

Na slici 4.11 je prikazan dio sheme za dekodiranje vrijednosti skeniranih uzoraka.

¹⁸ Indeks elementa koji je ulazna varijabla u ovaj blok označava bajtove počevši od 0 (prvi bajt nije 1. već 0) Isto vrijedi i za string.



Slika 4.11 Blok dijagram funkcije rfsSVI.vi

Izlazna varijabla je blok *Graph.vi*, a njemu potrebni elementi su početni offset ¹⁹, *m/bin*, i 8-bitno polje s uzorcima.

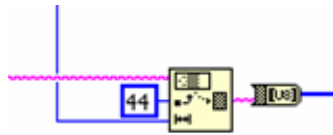
Postupak dekodiranja uzoraka ovisi o rezoluciji skeniranog niza. Za preispitivanje tipa koristi se *Condition Case.vi*. Na ulaz se dovodi rezolucija skeniranog niza kao logička varijabla (7).

U slučaju 4 bitnog uzorka izvršava se *False* dio uvjetnog bloka (kao na slici 4.10). Odvoji se dio stringa s uzorcima i pretvori u polje bajtova (vidi sliku 4.11). Kako jedan bajt sadrži dva podatka, kreirana je petlja (8) u kojoj se za svaki uzorak odvija razbijanje 8 bitnog zapisa u 4 bitni. To je izvedeno tako da sam zapis pretvorim u dva logička polja od 4 elementa. Transformacijom tih elemenata u 4-bitne brojeve dobio sam potreban oblik podatka. Na kraju se sve spoji u 8-bitno polje s vrijednostima veličine 4-bitna.

Parametar *m/bin* se računa tako da domet podijelim s brojem podataka. Broj 4 bitnih podataka je dvostruko manji od broja bajtova zapisanih uzoraka (*Data Bytes Count*).

¹⁹ Offset je 0, uzimamo u obzir sve elemente polja.

Dekodiranje 8 bitnog uzorka je puno jednostavnije: polje bajtova se samo proslijedi (vidi sliku 4.12).



Slika 4.12 Blok koji se nalazi pod *tru* opcijom

4.2.3 mtAlive

Sonarska glava u mirovanju²⁰ kontinuirano šalje mtAlive poruku u intervalima od 1Hz.

Tablica 4.8 mtAlive

40	30	30	30	38	08	00	FF	02	03	18	80	02	0A
Hdr '@'	Hex Length = 8 bytes				Bin Length = 8 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 3	mtS end BB'	Seq = End	Nde 02	LF

Pažnja: Sve vrijednosti mtAlive poruke nisu vjerodostojne onima koje su dane u dokumentaciji, npr 14. bajt bi za parametriran sonar trebao biti x00, a ne x80 što bi značilo da sonar nije parametriran.

4.2.4 mtSendData

Izgled poruke za zahtjev skeniranog niza prikazana je u tablici 4.9:

Tablica 4.9 mtSendData

40	30	30	30	43	0C	00	FF	02	07	19	80	02
Hdr '@'	Hex Length = 12 bytes				Bin Length = 12 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 7	mtS end D'ta	Seq = End	Nde 02
CA	64	B0	03	0A								
Current Time = 61891786 msec = 17:11:31.79				LF								

Parametar Current Time je varijable koja se koristi za sinkronizaciju s drugim uređajima. Kako mi tu opciju ne koristimo u mom slučaju ovaj parametar ima vrijednost x0000 0000.

²⁰ Pod mirovanjem podrazumijvam stanje kad ne očekivamo nikakav odgovor

4.2.5 Resetiranje sonara

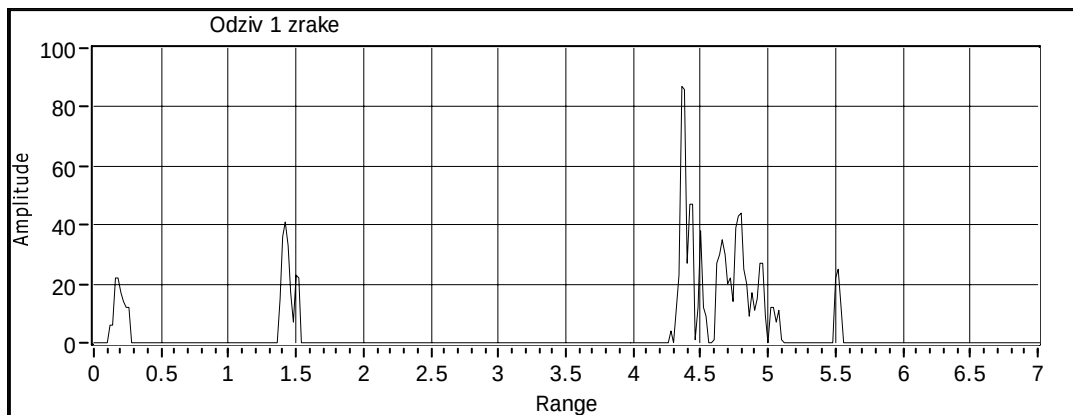
Sonar se može resetirati slanjem ‘mtReBoot’ naredbe. Ona je uvijek ista. Njezine vrijednosti su prikazani u tablici 4.10.

Tablica 4.10 mtReBoot

40	30	30	30	38	08	00	FF	02	03	10	80	02	0A
Hdr '@'	Hex Length = 8 bytes				Bin Length = 8 bytes		Tx Nde 255	Rx Nde 02	No. Byte = 3	mtR eBo ot	Seq = End	Nde 02	LF

4.3 Skenirani niz

Grafički prikaz skeniranog niza prikazan je na slici 4.13.



Slika 4.13 Skenirani niz

Ova skenirani niz je snimljen u bazenu pri *stare*²¹ modu rada u kružnom bazenu promjera 3.5 metra pri postavljenom dometu od 7 metara. Jednostavno je zaključiti da svi podaci na udaljenosti većoj od 3.5 metara su rezultat neželjene ponovne refleksije, pa imamo pojavu višestrukog ponavljanja signala: Skok na 4.5 jest refleksija prepreke na 1.25. Ta prepreka je najveća na prostoru bazena i ona označava udaljenost ruba. Skokovi koji se javljaju do oko 0.3 m je očitavanje same ronilice i sonara (šum).

Manje prepreke su slabije izražene i ne utječu znatno na skenirani niz (plutače na površini, kabel ronilice, otvori cijevi pumpe), no veće prepreke držači ljestvi i same ljestve često navode algoritam na pogrešan odabir mjerene udaljenosti.

²¹ Način rada u kojem je sonarska glava pozicionirana na jednom kutu

4.4 Određivanje udaljenosti

Na temelju skeniranog niza potrebno je eliminirati sve manje prepreke i odrediti udaljenost sonara od ruba bazena. Eksperimentalnim ispitivanjem pronađena su dva moguća rješenja:

- Za traženu udaljenost proglasi maksimalnu vrijednost amplitude koja se nalazi na udaljenosti manjoj od maksimalne dimenzije. Nedostaci su što je očitavanje šuma u nekim slučajevima veće od očitavanja ruba. Ako bi zanemarili prvih 0.3 metra ronilica bi bila neosjetljiva za toliko i efektivne dimenzije bazena bi se smanjile na vrijednost P' .

$$P' = (\text{radijus} - 0.3) \cdot 2\pi$$

Prednosti ovog postupka je jednostavnost.

- U početku se pretpostavi da su sve vrijednosti do 0.3 m rezultat šuma. Maksimalna vrijednost šuma se iskoristi za izračun tolerancije²² - najniže vrijednosti očitavanja koja se uzimam u obzir. Tolerancija se računa kao npr. 50% manji maksimalni šum. Prva sljedeća amplituda koja je veća od tolerancije jest potencijalni početak okoline tražene amplitude. Ako je maksimalna vrijednost te okoline veća od šuma onda je traženi rub pronađen, u suprotnom šum se proglasi rubom. Treba uračunati *offset* jer se eksperimentalno pokazalo da je tražena udaljenost na mjestu izdizanja. Ovim algoritmom je nemoguće prepoznati neželjenu refleksiju od same ronilice ako su ostale amplitude puno veće od amplitude ruba.

Implementirani algoritam je kombinacija ova dva, a njegov komentirani kod priložen je u dodatku A.

4.5 Određivanje pozicije

Dvo-dimenzionalnu poziciju opisujemo s dvije vrijednosti: x i y u Kartezijevom koordinatnom sustavu ili s r i φ u polarnom sustavu. Ovaj problem se svodi na rješavanje geometrijskog zadatka koji ovisi o prostoru u kojem se ronilica nalazi.

²² Tolerancija je potrebna jer moramo odrediti dio kad su očitavanja previše mala. U slučaju da je ronilica na dnu i parametar I_{gain} na minimalnoj vrijednosti moguća tolerancija iznosi 0, no kad ronilica 'lebdi' onda to nije uvijek slučaj.

Gdje je to moguće koristi se upravo kosinusov poučak iz razloga što je period ove funkcije od 0° do 180° , a to je ujedno i suma svih kuteva unutar trokuta čija pravila koristimo pri izračunavanju željenih vrijednosti.

Razlika kuteva $\angle D_1 D_2 S$ i $\angle D_1 D_2 T$ ekvivalentna je kutu α koji nam je nužan za izračunavanje željenog r -a (relacije od 4-3 do 4-6).

$$\angle D_1 D_2 S = \arccos\left(\frac{a^2 + R^2 - R^2}{2aR}\right) \quad 4-3$$

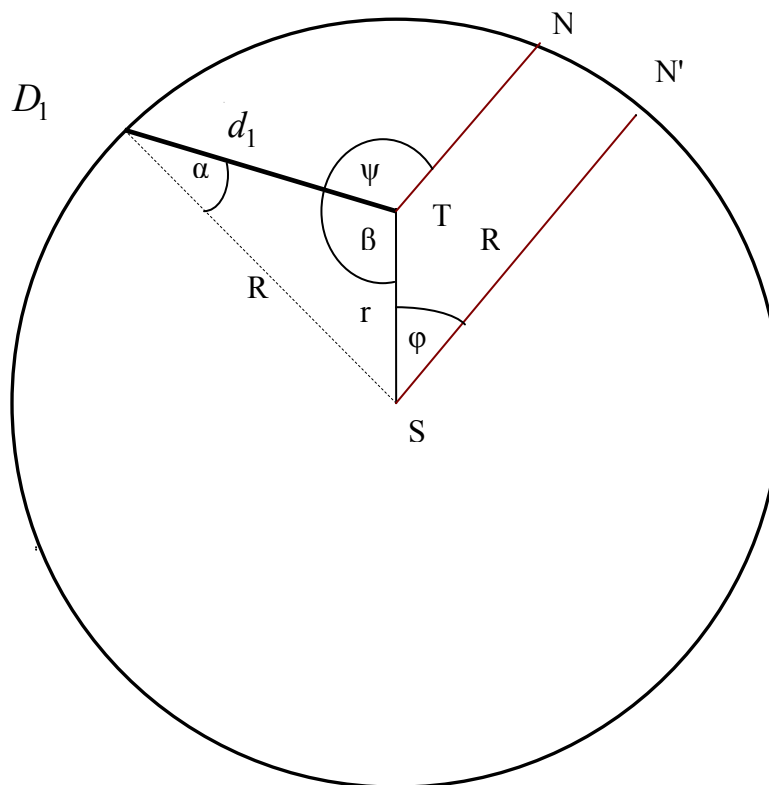
$$\angle D_1 D_2 T = \arccos\left(\frac{a^2 + d_2^2 - d_1^2}{2ad_2}\right) \quad 4-4$$

$$\alpha = \angle TD_2 S = \angle D_1 D_2 S - \angle D_1 D_2 T \quad 4-5$$

$$x = \sqrt{R^2 + d_2^2 - 2Rd_2 \cos(\alpha)} \quad 4-6$$

S određenim r -om znamo radijus kružnice na kojoj se nalazi ronilica, ali ne i φ (orijentacija u odnosu na kurs). Zato je potreban i kurs (vidi sliku 4.15).

Ψ – kurs : važno je napomenuti da se u ovom algoritmu koristi kurs kao orijentacija ronilice u odnosu na sjever, i podrazumijeva da vrijednost kursa raste suprotno kazaljke na satu kao i ostali kutevi. No u praksi kurs je pozitivno orijentiran, pa je potrebno unijeti transformaciju.



Slika 4.15 Određivanje orijentacije udaljenosti x u odnosu na sjever tj. duljinu SN'

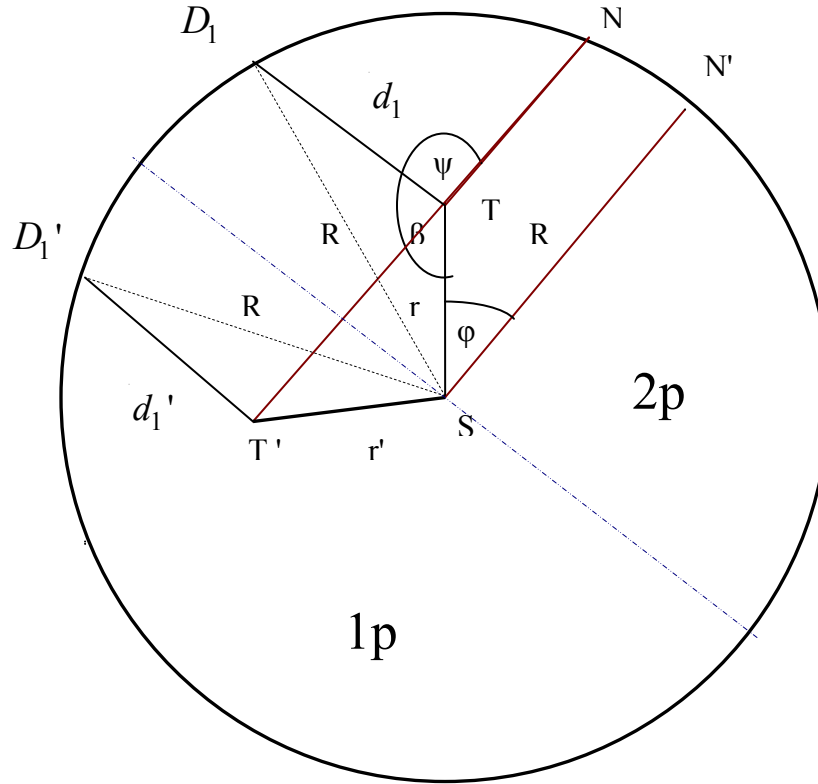
Iz slike se lako može uočiti da je kut $\angle NTS$ jednak $\pi - \varphi$, kut ψ je zadan, a $\angle STD_1$ izračunamo kosinusovim poučkom.

$$\angle STD_1 = \arccos\left(\frac{r^2 + d_1^2 - R^2}{2rd_1}\right) \quad 4-7$$

I sad imamo sve potrebne parametre:

$$\varphi = \psi - \pi + \angle STD_1 \quad 4-8$$

No iz slike 4.16 je vidljivo da zbog simetrije postoji još jedna točka T' unutar kružnice koja je jednako udaljena (d_1') i ima istu orijentaciju (ψ) u odnosu na sjever kao točka T .



Slika 4.16 Prikaz problema simetričnosti

Radi lakšeg razumijevanja kružnica se dijeli na dvije polutke $1p$ i $2p$ iscrtakanom linijom. Razlika polutke $1p$ i $2p$ jest različito računanje φ -a:

$1p$	$2p$	
$\varphi = \psi - \pi + \angle STD_1$	$\varphi = \pi - \angle STD_1 + \psi$	4-9

Ova razlika je neposredan rezultat činjenice da se orijentaciju ronilice računa pomoću trokuta tj pokušavamo prostor od 360° opisati unutar prostora od 180° , pa kad kut β prelazi granicu od 180° , trokut STD_1 prelazi u $SD_1'T'$ i kut φ sad obuhvaća jedan kut od trokuta $SD_1'T'$. Skup kuteva φ ne čini više raspon $180-360^\circ$ nego $0-180^\circ$. Ovo je logično ako uzmemo u obzir da prelazimo iz druge u prvu polutku.

$$\beta' = \pi - (\psi + \delta - \varphi) \quad 4-11$$

ξ' i ξ su uvijek manji od 90 stupnjeva, a veći od nula zbog toga jer je trokut unutar kružnice

$$\xi' < \frac{\pi}{2} \quad 4-12$$

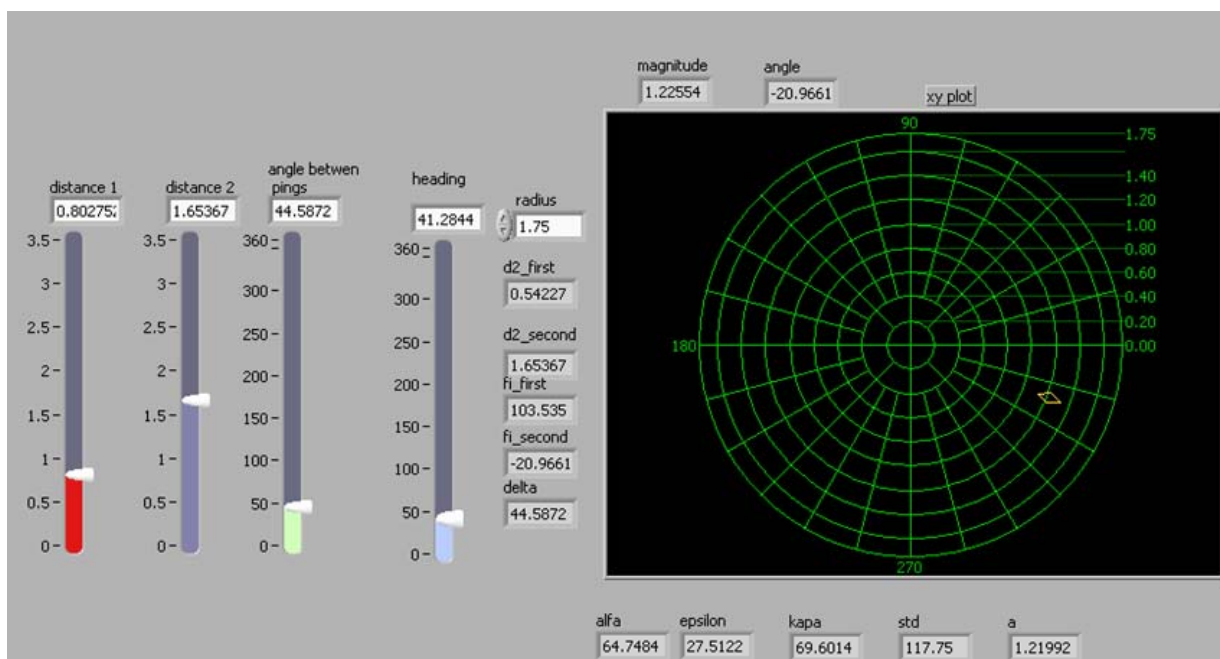
Pa je moguće koristiti i sinusov poučak

$$\xi' = \arcsin\left(r \frac{\sin(\beta')}{R}\right) \quad 4-13$$

$$\alpha' = \pi - (\beta' + \xi') \quad 4-14$$

$$d_2 = \sqrt{R^2 + r^2 - 2Rr \cos(\alpha')} \quad 4-15$$

Točnost ovih algoritama provjeren je s dva simulacijska podprograma. Jedan pronalazi dvije udaljenosti koje bi trebale predstavljati izlaze iz sonara za zadanu poziciju, kut δ i kurs ψ . Druga aplikacija određuje poziciju na osnovi dvije udaljenosti, kutu δ i kursu ψ .

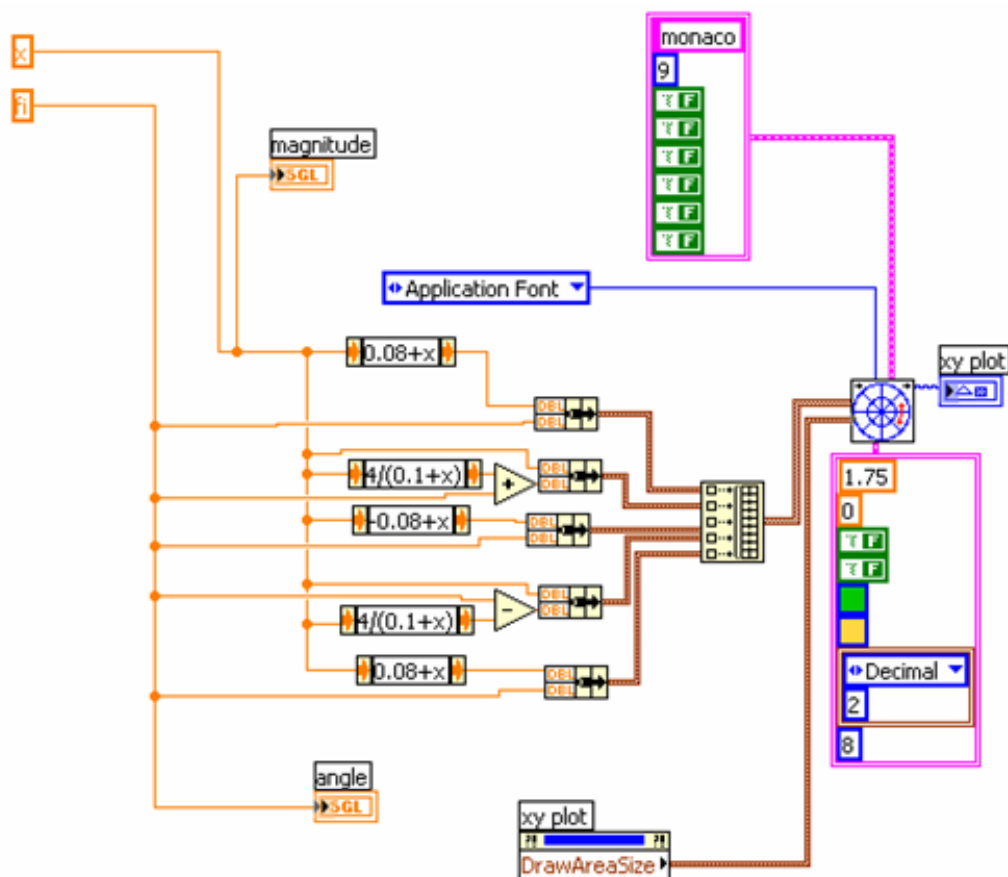


Slika 4.18 Sučelje simulativne aplikacije

Na slici 4.18 je prikazano sučelje aplikacije koji se koristi pri simulaciji algoritma. Nakon usavršavanja algoritma jedan gotovo identičan subVI objekt “*a4pcSVT*” (algorithm for position control SubVI) postavljen je kao temeljni dio aplikacije za lokalizaciju. Pri korištenju treba imati na umu da je kut φ kutni pomak u odnosu na sjever i to s orijentacijom suprotnom od kazaljke na satu.

Implementirani algoritmi za određivanje pozicije iz udaljenosti su priloženi i komentirani u dodatku B.

Na slici 4.19 prikazan je dio blok dijagrama. Pozicija je osim brojčanim vrijednostima prikazana i slikom. Sliku crta već gotov blok *PolarPlot.vi*. Bilo je potrebno promjeniti osnovne parametre za crtanje (širina grida, maksimalne, minimalne vrijednosti, stil slova itd). Veličina slike se prilagođava okviru zahvaljujući promjenjivim dimenzijama koje se učitavaju uz pomoć bloka *PropertyNode.vi*. Važno je napomenuti kako pozadina ne ovisi o bloku *PolarPlot.vi* što bi bilo logično već o *xy plot.vi*.



Slika 4.19 Blok dijagram funkcije koja crta ronilicu (kao točku)

Kako bi se na grafu vidjela samo jedna točka bilo je potrebno napraviti romb oko prave pozicije.

Iz točke pozicije odredio sam četiri točke koje čine četiri vrha romba, a kako je *Polar Plot.vi* izvorno blok za prikazivanje signala on te četiri točke međusobno poveže.

4.5.2 Grafički prikaz rotiranja ronilice ovisno o smjeru orijentacije

Da bi umjesto točke na grafu bio nacrtan prikaz ronilice potrebno je omogućiti rotiranje romba ovisno o kursu i pomak romba za vrijednost r -a. U tablici 4.11 su navedene koordinate točaka za prikaz ronilice u obliku romba s označenim vrhom koja se nalazi na ishodištu.

Tablica 4.11 Točke koje opisuju rubove prikaza ronilice

r	φ
0.09	0
0.1	140
-0.22	0
0.1	-140
0.09	0
0	0
0.12	0

Ove točke označavaju ronilicu na koordinatama (0,0). Da bi pomakli ronilicu u željeni položaj potrebo je najprije rotirati sustav za kut φ , a onda ga translirati za r .

Relacije transformacije u Kartezijev sustav:

$$\begin{aligned} x_{k0} &= r * \sin \varphi \\ y_{k0} &= r * \cos \varphi \end{aligned} \quad 4-16$$

Matrica rotacije

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad 4-17$$

Matrica translacije

$$\begin{pmatrix} x'_0 \\ y'_0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & x_{k0} \\ 0 & 1 & y_{k0} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r \\ y_r \\ 1 \end{pmatrix} \quad 4-18$$

Inverzna transformacija u polarne koordinate:

$$r'_0 = \sqrt{x'^0_0{}^2 + y'^0_0{}^2}$$

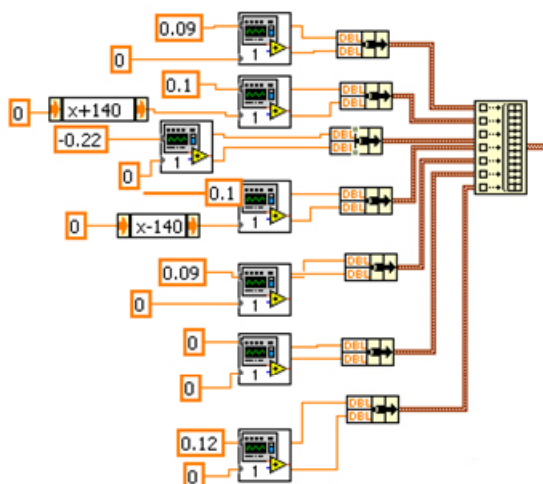
$$\varphi'_0 = \arctan\left(\frac{y'_0}{x'_0}\right),$$

$$\text{if}(x'_0 == 0) \rightarrow \varphi'_0 = 90^\circ$$

4-19

Izvršena je konverzija iz polarnog u kartezijev sustav, izvršene transformacije te je sustav ponovno vraćen u polarni. Kako u LabVIEW-u treba dodatni paket za rad s matricama izvedena je funkcija *PCTrans_Simulator* koja obavlja transformacije. Kod funkcije je priložen i komentiran u dodatku C.

Svaka točka ronilice provodi se kroz gore objašnjenje transformacije (vidi Sliku 4.20).



Slika 4.20 Blok dijagram funkcije koja crta ronilicu
(ronilica se okreće ovisno o kursu)

4.5.3 Kvadratni bazen

Za opis kvadratnih prostora koriste se koordinate u Kartezijevom sustavu $[x,y]$. Pokazalo se da je geometrijski puno teže opisati ovakav prostor stoga sam pribjegao pojednostavljenju uvođenjem još jedne varijable: *sa* (shift angle). Ovim načinom se znatno pojednostavljuje situacija jer znam orijentaciju ronilice u odnosu na x os, pa je moguće gotovo direktno izmjeriti x i y koordinate. Da bi se oni izmjerili potrebno je odgovarajuće

pozicionirati sonarsku glavu, a time prelazimo okvire geometrijskog problema i ulazimo u problem implementacije.

4.5.3.1 *Implementacija*

Kao i kod kružnog bazena moguće su dvije realizacije:

- pozicioniranje za svaku os (x i y)
- skeniranje prostora pod kutom od 180° .

Zbog potrebe velikog kuta ($\min 90^\circ + \text{abs}(sa)$) kojeg sonarska glava mora prijeći da bi izmjerila prikladne duljine isprva je vidljivo da će ovo pozicioniranje trajati znatno duže od prethodnog (kružni prostor). Iako skeniranjem ne možemo puno ubrzati proces pokazalo se da je ovaj način rada puno stabilniji i manje opterećuje rada sonar pa će u nastavku biti obrađena samo druga opcija.

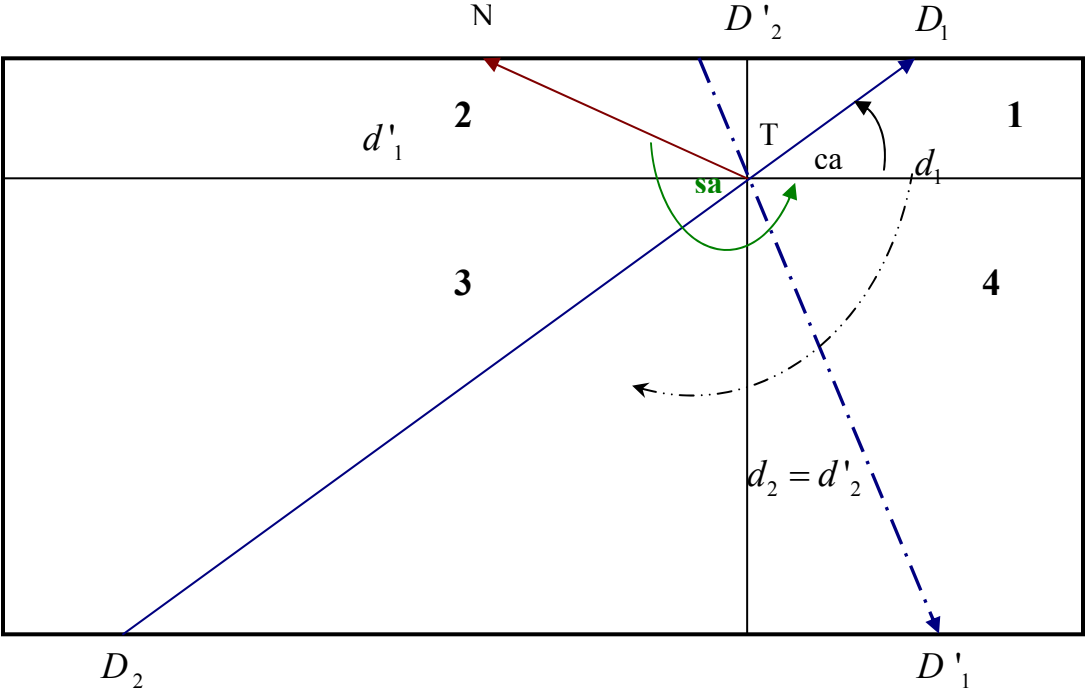
Prije samog i lokalizacije potrebno je ronilicu postaviti usporedno s jednim od rubova bazena. Odabrani rub se imenuje apcison koordinatnog sustava, a njezin pozitivan smjer jest u smjeru pramca ronilice²³. Time je ishodište koordinatnog sustava smješteno u jedan kut kvadra. Zapamtim orijentacijski kut kao vrijednost varijable sa . Iz ovog izvodimo još jednu pomoćnu varijablu pod nazivom ca (calibrated angle) i s njom označavamo pomak smjera kljuna ronilice u odnosu na apcisu. Ova varijabla prati pomak kursa, pa točnost ovog algoritma iznimno ovisi o ovom kutu. Iz tog razloga potrebno je kalibrirati položaj apcise prije svakog pokretanja aplikacije za pozicioniranje.

Glavna ideja je sljedeća: Skeniram prostor od 0° do 180° u smjeru kazaljke na satu (iscrtkana strelica na slici 4.21). Zahvaljujući kvadratnoj strukturi prostora, sonarska glava se bar jednom pozicionira u smjeru obe osi. Znamo duljino i širinu kvadrata i iz izmjerene udaljenost proračunamo koordinate.

U trenutku kalibracije $ca=0$, a $sa=hc$ ²⁴ (vidi sliku). Kalibraciju postizemo jednostavnim pritiskom na tipku *Calibrate angle* (vidi sliku 4.21) na kontrolnom sučelju.

²³ Pod pramcem podrazumijevamo prednji dio ronilice, no gore navedena pretpostavka vrijedi samo ako je *indikatorska lampica* sonara usmjerena u istom smjeru kao i trup ronilice. Ovaj postupak nazivam kalibracijom

²⁴ pozitivno orijentiran vrijednost kursa



Slika 4.21 Prikaz određivanja pozicije u kvadratnom bazenu

Pratim i bilježim promjenu kursa N tako da mijenjam vrijednos Ca prema relaciji 4-20:

$$ca = ca + (hc - sa) \quad 4-20$$

hc – heading course, kurs ,orijentacija u odnosu na sjever (smjer suprotan od kazaljke na satu).

sa – shift angel , kut za koji je smjer ronilice pomaknut u odnosu na pozitivnu x apcisu u trenutku kalibracije

Radi primjera pretpostavit ću da se ronilica nalazi u točki T kao na slici 4.21. Sjever je označen slovom N. Neka je $\overline{TD_1}$ duljina koju sonarska glava izmjeri pri kutu od 0° ²⁵. Tada je $\overline{TD_2}$ izmjerena duljina u krajnjem položaju skeniranog prostora (pod kutem od 180° , a duljine d_1 i d_2 su osi koje sonarska glava obuhvati na svom putu. Iz slike zaključujemo da je u ovom slučaju $x=x_{\text{max}}-d_1$, a $y=d_2$. No u drugom primjeru kad je ronilica usmjerena u točku D'_1 , relacije su nešto drugačije.

Ovaj problem rješavam na način da ishodište pomoćnog koordinatnog sustava postavim u točku T, i sad imam situaciju da za svaki kvadrant vrijedi zasebna relacija(od 4-21 do 4-24).

²⁵ To je kut između kljuna ronilice i početne pozicije sonarske glave(MP=3200)

Kvadrant prepoznajem ovisno o vrijednosti varijable ca :

I Kvadrant

Ako je $ca > 0^\circ$ & $ca < 90^\circ$

$$\begin{aligned} x &= x_{\text{max}} - d(\angle ca) \\ y &= d(90^\circ + \angle ca) \end{aligned} \quad 4-21$$

pri čemu je x_{max} = maksimalna dimenzija bazena u smjeru x osi (širina ili duljina),
 $d(\angle ca)$ = izmjerena duljina do ruba pri kutu ca .

$d(\text{ang}_x)$ duljina koja je skenirana u smjeru x osi.

ang_x – kut pri kojem je skenirana duljina u smjeru x osi

$d(\text{ang}_y)$ duljina koja je skenirana u smjeru y osi.

ang_y – kut pri kojem je skenirana duljina u smjeru y osi

II Kvadrant

Ako je $ca \geq 90^\circ$ & $ca \leq 180^\circ$

$$\begin{aligned} x &= x_{\text{max}} - d(\angle ca) \\ y &= y_{\text{max}} - d(\angle ca - 90^\circ) \end{aligned} \quad 4-22$$

y_{max} = maksimalna dimenzija bazena u smjeru y osi (širina ili duljina bazena),

III Kvadrant

Ako je $ca \leq -90^\circ$ & $ca \geq -180^\circ$

$$\begin{aligned} x &= d(180^\circ + \angle ca) \\ y &= y_{\text{max}} - d(270^\circ + \angle ca) \end{aligned} \quad 4-23$$

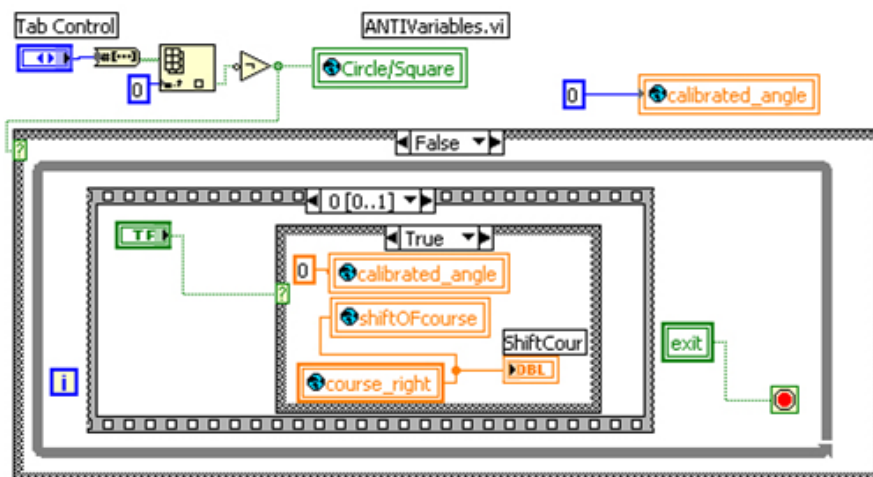
IV Kvadrant

Ako je $ca < 0^\circ$ & $ca \geq -90^\circ$

$$\begin{aligned} x &= d(90^\circ + \angle ca) \\ y &= d(180^\circ + \angle ca) \end{aligned} \quad 4-24$$

Pri tvorbi funkcija sam postupak određivanja pozicije u kvadratnom prostoru razbio sam na tri dijela:

1. **Praćenje pomaka kursa u odnosu na apcisu** izvedeno je u zasebnoj petlji. Ona se vrti unutar glavne aplikacije za pozicioniranje PC_PILOT_LOC.vi, neovisno o glavnoj petlji. Sve vrijednosti razmjenjuje preko globalnih varijabli. No, da bi se petlja vrtila treba biti ispunjen uvjet koji ovisi o položaju *Tab_Control.vi*. Unutar petlje nalazi se *Stacked Sequenced Block.vi* namijenjen za slijedno odvijanje operacija: Prvo se provjerava aktivnost tipke za kalibraciju *Angle Calibration* (Calibrate_angle.vi) te ovisno o stanju izvrši kalibracija, a potom se vrši promjena varijable *ca* s obzirom na pomak kursa.



Slika 4.22 Praćenje pomaka kursa u odnosu na apcisu

2. **Algoritam za određivanje kvadranta i kuteva s kojih se mjere udaljenosti** nalazi se unutar aplikacije za skeniranje²⁶ i to u bloku PC_DIST_SEL.vi.

- Kut *ca* se postavlja u interval $[-180, 180]$.
- Određuje se kvadrant i kutevi ang_x i ang_y ²⁷ prema relacijama 4-21, 4-22, 4-23 i 4-24.
- Dobiveni kutevi se pretvaraju u poziciju motora prema relaciji 4-25, a zatim uspoređuju s trenutnom pozicijom. U slučaju jednakosti izmjerena udaljenost se zapisuje u varijablu *d1* ili *d2* ovisno o *x* ili *y* koordinati.

²⁶ Vidi poglavlje 5

²⁷ *ang_x* – kut pri kojem je skenirana duljina u smjeru *x* osi
ang_y – kut pri kojem je skenirana duljina u smjeru *y* osi

Važno je uočiti da u relaciji 4-25 postoji približno zaokruživanje zbog činjenice da se sonarska glava giba korakom od 1.8° , pa se uzima približne vrijednosti (ako je to potrebno).

- Specijalni slučaj je pozicija 6400 koja ne postoji već je to početak novog kruga, pa je ekvivalentan 0.

Programski kod algoritma naveden je u dodatku D.

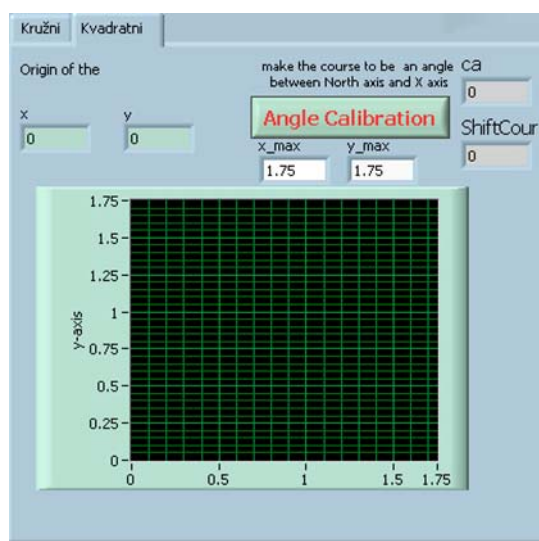
$$\begin{aligned} ang_x &= \text{ceil}(ang_x/1.8) * 32) + MPleft \\ ang_y &= \text{ceil}(ang_y/1.8) * 32) + MPleft \end{aligned} \quad 4-25$$

3. Algoritam za određivanje x i y koordinata iz vrijednosti varijabli (Blok *PC_SQUARE.vi*):

- broj kvadranta (*case*)
- vrijednosti udaljenosti za kut ang_x i ang_y (d1 id2)
- x_max i y_max

Ovaj blok ujedno i crta točku u *xy graph.vi* funkciji.

Sučelje sam realizirao kao dio multiopcionalne aplikacije (vidi sliku 4.23).



Slika 4.23 Sučelje za kvadratni bazen

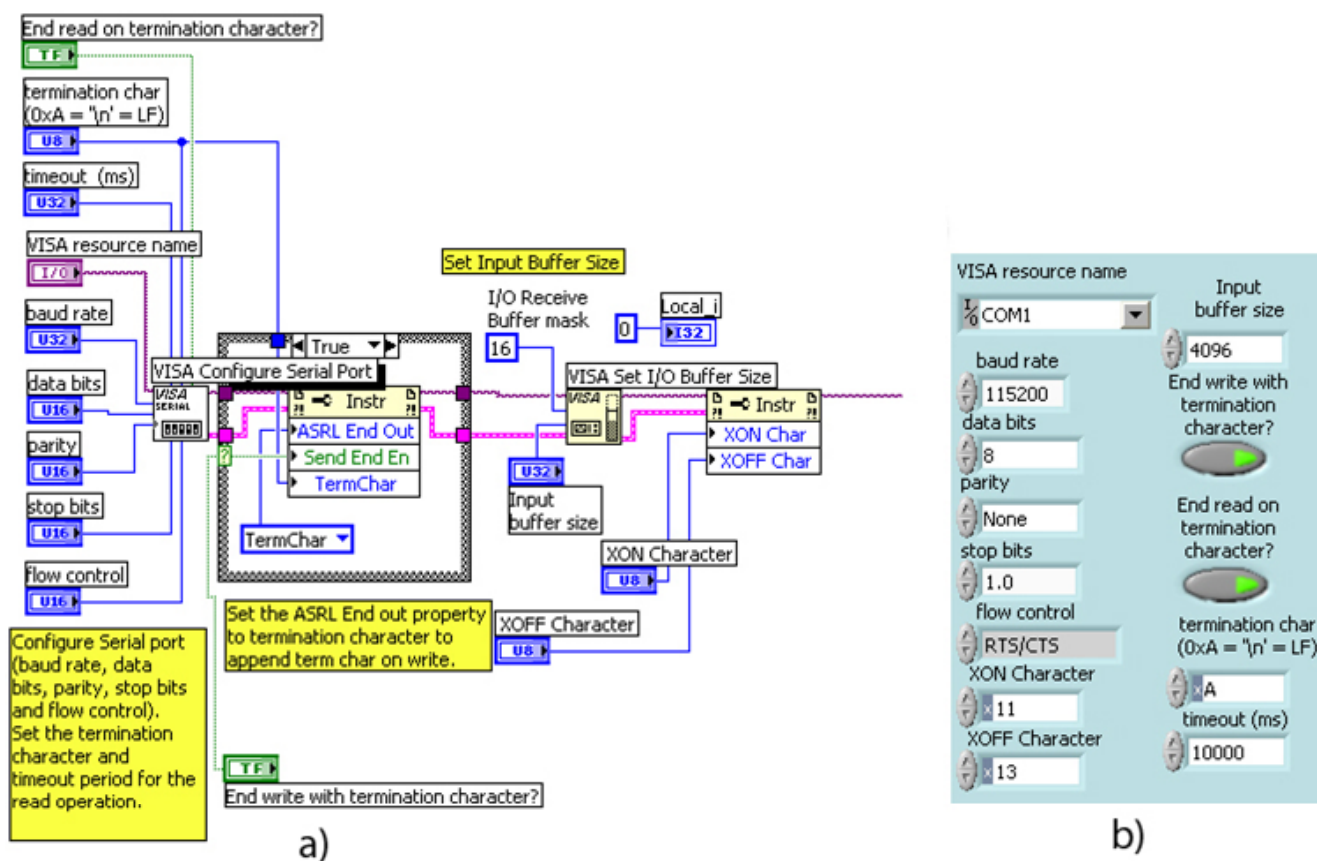
4.6 Aplikacije za lokalizaciju

Sve ove aplikacije imaju zajedničke karakteristike. Realizirane su tako da postoji jedna glavna petlja unutar koje se sve odvija. Petlja se vrti dok aplikacija radi, a veza sa PC_pilotom su globalne varijable. Zbog problema s radom dva porta s različitim brzinama bilo je nužno realizirati ovu aplikaciju kao zasebnu i potrebno ju je pokretati **nakon** PC pilota. PC-pilot koristimo u svrhu upravljanja ronilicom, te dohvat vrijednosti kursa ronilice.

4.6.1 Čitanje i pisanje s porta

Komunikacija aplikacije sa sonarom zahtjeva čitanje i pisanje sa RS232 porta. Iskoristio sam već gotove blokove : *Visa Configure Serial Port.vi* , *Visa Set I/O Buffer Size.vi* , *Visa Write.vi* , *Visa Read.vi*.

Port se konfigurira samo jednom i to prije ulaska u glavnu petlju. Vidi sliku 4.24 a) .



Slika 4.24 a) Blok dijagram za komunikaciju preko porta b) sučelje

Vrijednost parametar su definirane od strane sonara (vidi sliku 4.24 b)).

4.6.2 Spremanje podataka (logiranje)

Proračunate podatke je nužno sačuvati i spremiti na neku memorijsku lokaciju jer oni čine temelj povratne grane u zatvorenom upravljačkom krugu, čije projektiranje tek slijedi.

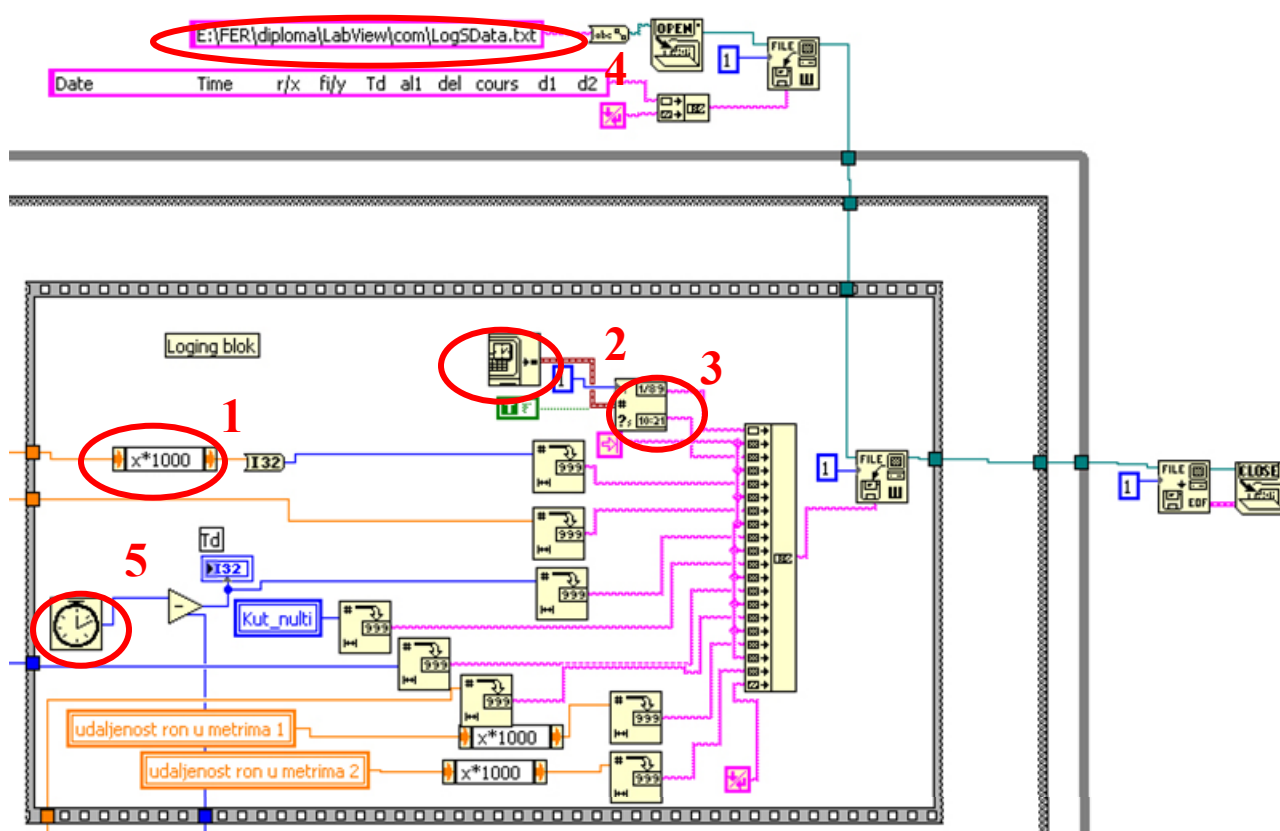
Osim za direktno upravljanje, informacija o poziciji je namijenjena u svrhu reproduciranja trajektorije, stoga je bilo neizbježno dizajnirati blok koji prilikom rada aplikacije zapisuje svaku proračunatu poziciju i neke važne parametre.

Vrijednosti koji se zapisuju prikazani su u tablici 4.12.

Tablica 4.12 Podaci koji se zapisuju u txt file

Date	Datum zapisivanja
Time	Vrijeme zapisivanja
r	Udaljenost r od ishodišta (polarne koordinate)
x	Koordinata x (Kartezijev sustav)
ϕ	Kut ϕ u polarnim koordinatama
y	Koordinata y (Kartezijev sustav)
Td	Ukupno vrijeme potrebno za proračun pozicije
α_1	Kut α_1 je početni kut skeniranja /
δ	Kut δ između dva zvučna signala, odnosno dvije udaljenosti
course	Orijentacija ronilice u odnosu na sjever
d1	Očitana udaljenost ronilice i ruba u smjeru pod kutem α_1
d2	Očitana udaljenost ronilice i registriranog ruba u smjeru pod kutom $\alpha_1 + \delta$

Implementacije gore navedenog logiranja prikazana je na slici 4.25.



Slika 4.25 Blok dijagram logiranja

U bloku za sekvencijalno izvršavanje funkcija *Flat Sequence Structure* potrebni podaci se transformiraju kako bi se mogli zapisati u string. Zbog pojednostavljenja sve su cijeli brojevi pa se pozicija, i udaljenosti zapisuju u milimetrima. (1). Datum i vrijeme se izvlači i transformira pomoću posebnih blokova *Get Date/Time In Seconds* (2) i *Get Date/Time To String* (3).

Parametar *Td* računam tako da od vremena učitano prije postupaka lokalizacije oduzmem vrijeme koje je program učitao posljednje (5)

Da bi izvršili upis potrebno je otvoriti datoteku (*Open File*), upisati odgovarajuću adresu i ime datoteke (4), a nakon zapisivanja datoteku zatvaramo.

Prije samog zapisivanja zapisuju se i nazivi samih parametara pa je logiranje strukturirano u obliku tablice:

To se vidi na primjeru iz datoteke `LogSDData.txt`

Date	Time	r	fi	Td	al1	del	kurs	d1	d2
Tuesday, June 06, 2006	8:34:44 PM	675	384	1070	0	315	308	1460	2000
Tuesday, June 06, 2006	8:34:45 PM	410	370	1068	0	315	320	1460	1740

Tuesday, June 06, 2006	8:34:46 PM	90	308	1064	0	315	268	1680	1740
Tuesday, June 06, 2006	8:34:47 PM	90	168	1092	0	315	208	1680	1660
Tuesday, June 06, 2006	8:34:48 PM	632	298	1072	0	315	250	1260	1660
Tuesday, June 06, 2006	8:34:50 PM	496	286	1068	0	315	276	1260	1420
Tuesday, June 06, 2006	8:34:51 PM	650	317	1066	0	315	292	1140	1420
Tuesday, June 06, 2006	8:34:52 PM	610	307	1118	0	315	304	1140	1280
Tuesday, June 06, 2006	8:34:53 PM	610	301	1061	0	315	298	1140	1280

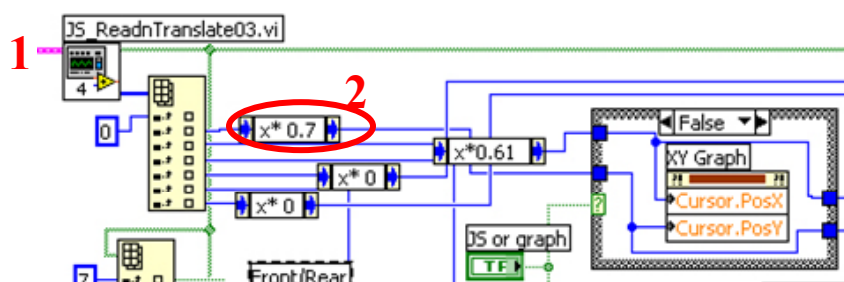
U ovom su dijelici zapisa registrirane pozicije pri kretanje ronilice iz čega se vrlo jednostavno rekonstruira trajektorija.

4.6.3 Podešavanje brzine gibanja ronilice u svrhu lokalizacije ronilice

Kod online postupaka jako je bitna brzina skeniranja uzoraka tj. vrijeme diskretizacije (T_d). Najmanje moguće vrijeme proračuna²⁸ primjerenih vrijednosti pozicije jest oko 600 ms, a ronilica se u normalnom radu giba brzinom od 1 m/s. Tu se javlja problem jer je za precizna očitavanja nužno da ronilica miruje dok se ne očitaju parametri. Ovaj nedostatak pokušat će se zaobići smanjenjem brzine gibanja ronilice. Brzina mora biti toliko mala da je pomak koji napravi u 600 ms gotovo zanemariv za proračun pozicije. Naravno ovaj postupak unosi veliku pogrešku jer je kut između udaljenosti mali, pa se zahtijeva velika preciznost. O tome će se više posvetiti eksperimentalni dio ovog rada.

Snaga propulzora u PC Pilotu se mijenja unutar skaliranog intervala $[-32768, +32768]$ zasebno za svaku os gibanja, pa je zadatak bio smanjiti taj interval na dovoljno male vrijednosti.

Brzina gibanja ronilice se razlikuje ovisno o smjeru gibanju, pa su i koeficijenti za smanjivanje snage propulzora različiti (vidi sliku 4.26).



Slika 4.26 Dio blok dijagrama PC-pilota gdje je potrebno uvesti skaliranje

²⁸ Proračun u ovom kao i u sličnom kontekstu uključuje sve potrebno za dobivanje pozicije što uključuje skeniranje, komunikacija sa sonarom itd.

Ulazna varijabla (1) je string sa 7 elemenata. Zadnji šest su skalirane vrijednosti snage propulzora za 6 poluosi gibanja. Vrijednost koeficijenta određena je eksperimentalno, a prikazani su redom kao u tablici 4.13.

Tablica 4.13 Vrijedosti koeficijenata za skaliranje

k	koeficijenti kojim se množi vrijednost u svrhu smanjenja brzine
0.7	+ x os – gibanje naprijed (2)
0.61	- x os – gibanje unatrag
0	+ y os – gibanje udesno
0	- y os – gibanje ulijevo
Brzinu uranjanja i izdizanja nisam modificirao.	

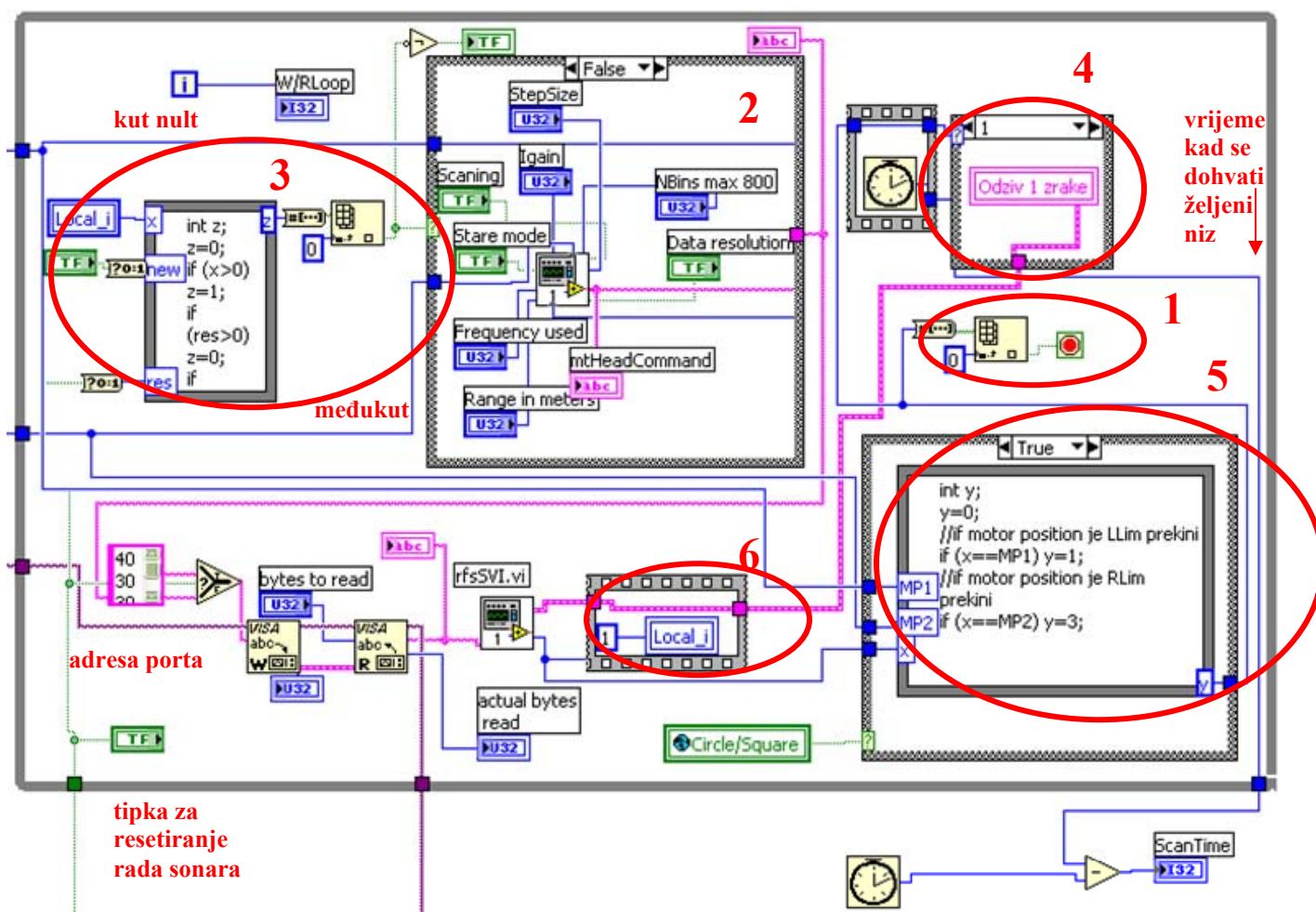
4.6.4 Aplikacija temeljena na skeniranju

Temeljni princip rada:

Pri pokretanju aplikacije, sonaru se jednom pošalju parametri za skeniranje, te počne postupak skeniranja²⁹. Skenira se određeno područje najvećim mogućim korakom (1.8). Pri graničnom kutu skeniranog sektora učitani niz se zapiše u memoriju. Dotični niz sadrži informaciju o udaljenosti, a on se dalje koristi u funkcijama lokalizacije.

Na slici 4.27 je prikazana implementacija bloka koji razlikuje ovaj način rada od ostalih.

²⁹ Šaljem poruku za zahtjev informacije, te primam poruku koja sadrži skenirani niz (informaciju o udaljenosti) i ostale parametre (npr. trenutna pozicija)



Slika 4.27 Blok Dijagram aplikacije koja se temelji na skeniranju

Na slici 4.27 su označeni linijski ulazi i izlazi iz petlje u kojoj je implementiran postupak skeniranja.

Vidljivi ulazi u petlju su:

kut nulti – kut je zapisan kao pozicija motora sonarske glave (0 – 6399)

međukut prvi - kut je zapisan kao pozicija motora sonarske glave (0 – 6399)

vrijednost tipke za resetiranje signala (1- reset, 0 – normal)

Vidljivi izlazi iz petlje su:

vrijeme kad se dohvati željeni niz – iako se niz mijenja kod svakog pomaka sonarske glave iz petlje izlazi samo ona krajnja vrijednost odnosno vrijeme u trenutku zapisivanja u memoriju, jer se jedino tad petlja prekida (1).

Blok za kodiranje poruke nalazi su u uvjetnom bloku (2), a uvjeti da se on izvrši su:

- prva iteracija glavne petlje ($Local_i == 0$) , Prije ulaska u glavnu petlju varijabla $Local_i$ se postavlja u 0, a nakon svakog čitanje s porta u 1 (4).
- resetiranje sonara ($REBOOT == 1$). Sonaru se šalje poruka za resetiranje.
- učitavanja novih parametara ($Scanning\ changes == 1$). Tipka *Apply*

Ako nijedan od navedenih uvjeta nije ispunjen odvija se skeniranje: u uvjetnom bloku se pod false nalazi string sa `mtSend` porukom.

String se iz uvjetnog bloka prosljeđuje na blok za pisanje u sonar. Slijedi čitanje, dekodiranje i zapis u memoriju. No zapisuju se samo oni koji ispune uvjet (4): sonarska glava mora biti pozicionirana na željeni kut. Kod kružnog bazena zapisuju se granični kutevi skeniranja, a kod kvadratnog bazena kutevi se računaju³⁰. Iz tog razloga postoji još jedan uvjetni blok (5) u kojem funkcija registrira da li je sonarska glava trenutno pozicionirana na željenom kutu.

Ovisno o tom da li se radi o kružnom ili kvadratnom bazenu (npr. `Circle/Square=true` → kružni bazen) aktiviraju se odgovarajuće funkcije. One zapišu odgovarajući niz u odgovarajuću memoriju tj lokalnu varijablu.

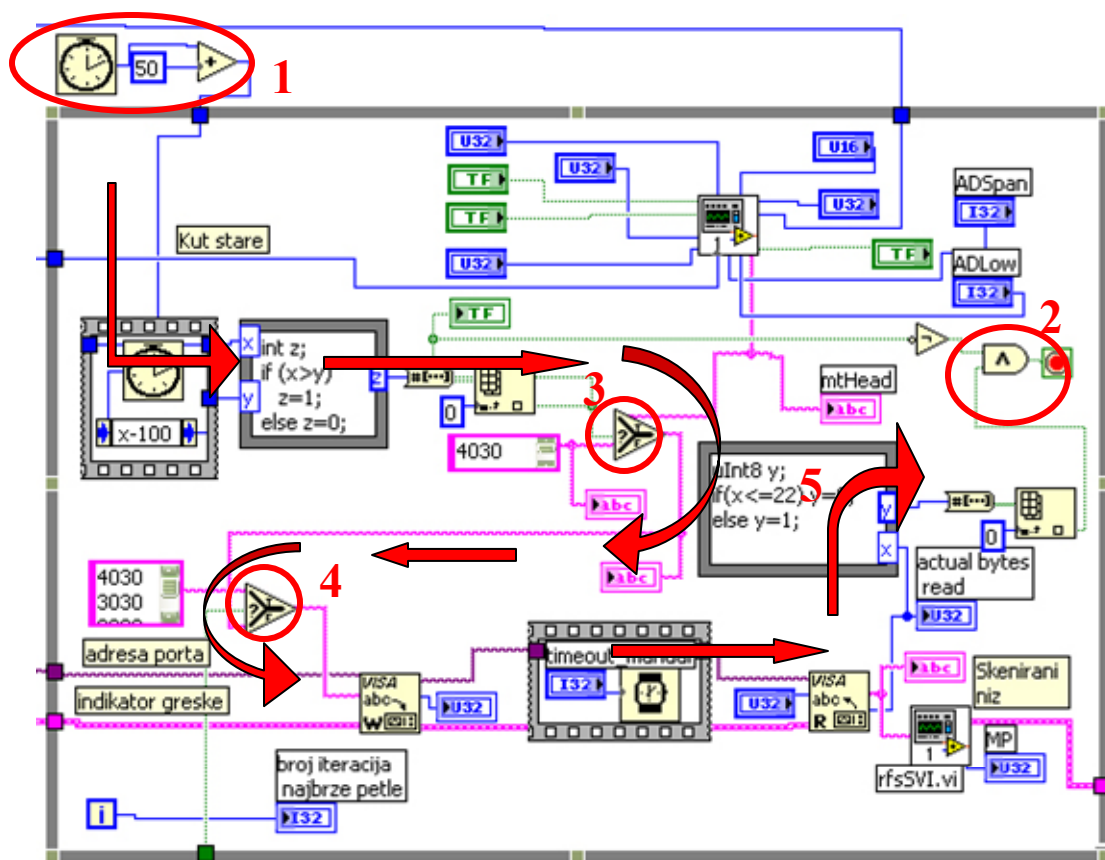
Kako su za lokalizaciju potrebne dvije udaljenosti na različitim pozicijama potrebne su dvije lokalne varijable u koju se upisuju dva različita kuta (kružni bazen).

4.6.5 Aplikacija temeljena na pozicioniranju

Sonar nije originalno namijenjen pozicioniranju (*stare mode*) i ne preporuča se njeno korištenje. Ipak realizirana je aplikacija koja se temelji na pozicioniranju i eksperimentalno je ustanovljeno da ima nezaobilaznih nedostataka.

Temeljni princip rada je sljedeći: sonar radi u *stare* načinu rada odnosno za svaku udaljenost se postavljaju novi parametri. Čeka se dok sonarska glava ne pozicionira na željeni kut, a onda jednostavno učitam poruku sa skeniranim nizom.

³⁰ Postupak računanja je objašnjen u prethodnom poglavlju



Slika 4.28 Blok dijagram aplikacije koja se temelji na pozicioniranju

Na slici 4.28 je prikazan dio blok dijagrama odgovoran za komunikaciju s sonarom: u prvoj iteraciji petlje sonaru se šalju parametri za lokalizaciju, a u drugoj zahtjev za podacima o udaljenosti i čita dobivena poruka. Ako se dobije željena poruka petlja se prekida, a skenirani niz šalje na daljnju obradu. Pri jednoj iteraciji glavne petlje, ova petlja se proziva dva puta iz razloga jer je za proračun pozicije potrebna vrijednost udaljenosti za dva različita kuta.

Tok dijagram opisan je strelicama na slici. Uporedbom vremena prije pokretanja petlje (1) i vremena u trenutku iteracije odlučuje se o sadržaju poslanih poruka:

- Ako je vrijeme pokretanja petlje uvećano za 50ms veće od vremena u trenutku iteracije (smanjeno za 100ms) tada se kroz blok *Select.vi* dalje prosljeđuje kodirana poruka s parametrima (mtHeadData), a na stop blok (2) jedan od uvjeta za pokretanje sljedeće iteracije.
- U suprotnom slučaju prosljeđuje se poruka mtSendData.

Nakon provjere zahtjeva za resetiranje (4) poruka se piše na port. Nakon proizvoljnog čekanja³¹ čitamo poruku s porta, provjeravamo o kakvoj je poruci riječ na temelju njezine duljine(5):

- Poruka jednaka ili manja od 22 bajta je znak da treba poslati *mtSendData* poruka. Ova usporedba je nužna jer se događa da se iz prve ne uspije pročitati *mtHeadData* poruka, već je potrebno provesti još nekoliko iteracija.
- Kad se *mtHeadData* poruka uspješno pročita, petlja se prekida (2), pa dekodirani skenirani niz izlazi iz petlje.

³¹ U normalnom radu čekanje je vrlo malo , od 0 do 10ms, no ova opcija je naimjenjena isključivo u svrhu testiranja

5 REZULTATI EKSPERIMENTALNIH ISPITIVANJA

Ispitivanja su izvršena u kružnom laboratorijskom bazenu Fakulteta elektrotehnike i računarstva (vidi sliku 5.1). Dimenzije bazena iznosa:

- visina 1.21 m
- promjer 3.5 m

Karakteristike Bazena su sljedeće:

Dubina vode u trenutku ispitivanja je iznosila 1.05 m.

Na južnoj strani bazena uronjene su stepenice.

Sa sjevero-zapadne strane postoje dvije šupljine malo ispod razine vode. To su otvori za pumpu koja filtrira vodu. Izlazni otvor je veći i ono više utječe na skenirani niz

Bazen je građen od aluminijskog lima obložen plastikom.



Slika 5.1 Laboratorijski bazen

Kabel ronilice, plutača s klorom, stepenice i otvori utječu na izgled skeniranog niza, i u slučaju lokaliziranja predstavljaju smetnju. U većini slučaja amplitude odbijenih signala su zanemarivo manje u odnosu na amplitudu ruba, te ih algoritam zaobilazi.

Stepenice najviše utječu na mjerenja, no ako je ronilica dovoljno udaljena (oko 1.5 m) očitavanja su ipak točna.

5.1 Usporedba brzina različitih aplikacija

Pri mjerenju korišteni parametri :

$\delta=45^\circ$, $Nbins=200$, $Range=4$ m, $Kut_multi=0^\circ$, $Kut_drugi=45^\circ$, $Data\ Resolution=8$ bita, $Igain=0$;

5.1.1 Aplikacija temeljena na pozicioniranju

Ukupno vrijeme potrebno za lokalizaciju iznosi: 2389 ± 50 ms³²

1 iteracija vanjske petlje: 1176 ms

2 iteracija vanjske petlje: 986 ms

Sistemska kašnjenje koje je jednako za obe aplikacije iznosi 200 ± 30 ms.

Tako velika vremenska konstana posljedica je činjenice da izvršenje `mtHeadCommand` naredbe zahtjeva vrijeme od 1s.

Kako bi to potvrdio završio sam eksperimente na osnovnom modelu za čitanje i pisanje ('*Atesting.vi*') koji se temelji na osnovnom pozicioniranju. Postavljena su dva brojila: jedan za pisanje a drugi za čitanje na potr. Potrebna vremena u slučaju da je sonarska glava uvijek pozicionirana na jednom kutu iznose:

pisanje `mtHeadCommand` 6 ms

čitanje `mtAlive` 959 ms

pisanje `mtSendData` 2 ms

čitanje `mtHeadData` 26 ms

Iz gore navedenih podataka vidljivo je da najduže čekamo čitanje `mtAlive` poruke iz čega se da zaključiti da sonar najsporije obradi poruku `mtHeadCommand`.

³² Razlog ovako velikog vremena bit će objašnjen u nastavku

Proces Reboot je poseban slučaj koji traje oko 11300 ms. Ipak ovo je nužno u slučaju nefunkcionalnosti sonara.

5.1.2 Aplikacija temeljena na skeniranju

Kod ovog načina rada sonarska glava se pomiče između željenih kutova tako da skenira. Skeniranje je vrlo brz proces. Sonarska glava se može kretati maksimalnim korakom od 1.8 stupnjeva, pa odabiremo upravo tu opciju.

Vrijeme T_s koje je teoretski potrebno za pozicioniranje sonarske glave od jednog do drugog kuta iznosi prema relaciji 5-1.

$$T_s = \frac{\delta}{step_size} \cdot (Tr + Tw) \quad 5-1$$

pri čemu je

Tr – vrijeme potrebno za čitanje `mtHeadData` poruke

Tw – vrijeme potrebno za pisanje poruke `mtSendData`

δ – pomak između dvije udaljenosti

$step_size$ – korak motora pri kojem sonarska glava skenira.

Tu nije uračunato vrijeme potrebno za izvršavanje `mtHeadCommand` i `mtAlive`, što se događa samo jednom, pri inicijalizaciji. Sonar pri skeniranju automatski vrši pozicioniranje: kad odgovori na `mtSendData` poruku s `mtHeadData` porukom, sam se pomakne za jedan korak (*step size*) naprijed, a na kraju sektora obavlja promjenu smjera gibanja.

To znači da bi pri normalnom radu za parametre $\delta = 25^\circ$, $step_size = 1.8^\circ$ i $Nbins = 200$. T_s teoretski iznosio 389 ms.

No to nije slučaj jer imamo sistemsko kašnjenje oko 227 ms zbog neotkrivenih razloga, pa T_s varira oko 591 ms. Svi ostali proračuni traju jednu do dvije ms i to je još jedan od dokaza da je mjerenje udaljenosti skenirajućim sonarom usko grlo u postupku lokalizacije.

5.1.3 Zaključak usporednog testa

Aplikacija temeljena na skeniranju ima puno bolja svojstva:

1. Neusporedivo brža vremena
2. Znatno manje opterećuje sonarsku glavu
3. Stabilan rad, dok je aplikacija temeljena na pozicioniranju nestabilna - Tritech stručnjaci ne preporučuju *pozicionirajući*³³ način rada sonara. Pri ispitivanju je ustanovljeno da postoji velika mogućnost od pada programskog sustava u sonaru pri prelasku iz jedne aplikacije u drugu.

Iz tog razloga u daljnjim eksperimentima koristit ću aplikaciju temeljena na skeniranju.

5.2 Eksperimentalno ispitivanje mogućnosti sonara

U ove svrhe koristi se aplikacija temeljena na skeniranju. Kako su za određivanje pozicije najbitnija svojstva točnost i preciznost izmjerene udaljenosti, one se i posebno razmatraju. Za online lokalizaciju bitna je još i brzina.

5.2.1 Utjecaj vremena diskretizacije skeniranog niza na brzinu lokalizacije

Mijenjanjem broja uzoraka (NBins) također mijenjamo i vrijeme obrade odnosno vrijeme potrebno da se pročita mtHeadData poruka. Negativna posljedica ovog jest da smanjujemo preciznost.

Rezultati eksperimenta:

NBins:	50	100	200	300	400	500	600	700
Tw:	2	2	2	2	2	2	2	2
Tr:	10	16	26	35	45	50	50	ne može pročitati s porta

Tr – vrijeme potrebno za čitanje mtHeadData poruke

Tw – vrijeme potrebno za pisanje poruke mtSendData

Ispitivanjem je utvrđeno da je najbolje koristiti NBins u intervalu od 100 do 200.

³³ eng. stare mode

Mjerenja izvršena pri vrijednosti NBin-a od 200 ms.

5.2.2 Utjecaj dometa (*range*) na brzinu

Mijenjanjem dometa mijenja se vrijeme potrebno za čitanje mtHeadData npr.:

range	potrebno vrijeme
4 m	26 ms
50 m	70 ms
100 m	132 ms

Uzorci skeniranog niza mogu biti 4- bitne ili 8- bitne veličine. Pri 4 bitnim uzorcima poruka je kraća i za naslutiti jest da je i komunikacija brža. No, ovdje to nije slučaj.

5.2.3 Utjecaj brzine komunikacije sonar - računalo na brzinu aplikacije

Baud rate – brzina prijenosa poruke izražena u jedinici [bit/s]

Rezultati ispitivanja prikazani su u tablici:

Baud rate	t2r [ms]	t2w [ms]	actual byte read[bit/s]
9600	NEKOLIKO REDOVA VELIČINE SPORIJE		242-243/244
57 600	48	3	243/244
115 200	26	2	244/244
153600	NE RADI		

Vidljivo je da kod manjih brzina susrećemo pojavu gubitaka dijela poruke. Najprikladnija brzina prijenosa iznosi 115 200 [bit/s].

5.2.4 Utjecaj skeniranog niza na točnost lokalizacije

Dobro prepoznavanje skeniranog niza nužni je preduvjet za dobru lokalizaciju. Parametar *Igain* postavljam na 0.

Frekvencija zvučnog signala ne može se mijenjati već je ona zadana u „firmware“ programu sonarske glave i iznosi 700 kHz s 100 kHz širine pojasa *chirp* signala. Ovo podešenje daje najbolje rezultate. Implementirani algoritam jako ovisi o ovom parametru. Kompanija Tritech ne isporučuje program za mijenjanje ovih parametara i uskraćuje dodatnu dokumentaciju o „firmware“ programu. U slučaju resetiranja na tvorničke vrijednosti potrebno se izravno obratiti Tritech-u. A postojeća opcija za mijenjanje frekvencije koja je čak ponuđena i u originalnom software-u je samo varka.

Pri frekvencije od 325 kHz šum je puno veće amplitude, a očitavanja ruba puno slabija.

Originalno podešenje sonara je bilo 725 kHz s 50 kHz širine pojasa *chirp* signala.

Pomak udaljenosti (*offset*) korišten pri mjerenju iznosi 0.11 m, no pokazalo se da treba iznositi 0.1 m. Maksimalna amplituda oko prave udaljenosti nije tražena udaljenost od ruba već je potrebno uzeti prvu najbližu zapaženu vrijednost veću od nule. Pokazalo se da ta vrijednost puno više varira u odnosu na maksimalnu vrijednost pa se koristi negativni pomak (*offset*) u odnosu na maksimalnu vrijednost.

Sonar na suhom očitava samo šum.

Promatranjem skeniranih nizova u različitim pozicijama ronilice opazio sam određeno pravilo izdizanja amplituda u skeniranom nizu:

- Uvijek postoji šum koji doseže negdje do 0.3 metra. Vrijednosti amplituda variraju od 20 do 40, ako je skalirani interval jednak 255, i ako se ronilica nalazi na dnu. Šum postoji jer ronilica 'čuje' samu sebe. Kad je ronilica blizu površine intenzitet odbijanja je nešto veći zbog odbijanja od površine (prijelaz voda – zrak)
- Amplituda ruba bazena mijenja se ovisno o tome koliko je intenzitet odbijanja zvučnog signala. Udaljavanjem od ruba, intenzitet zvučnog signala opada, pa se i amplituda postepeno približava razini šuma, no u većini slučajeva je ipak veća. Udaljenost ruba se ne nalazi na mjestu maksimalne amplitude već na početku izdizanja.
- Zbog fizikalnih svojstava zvučnog signala neželjene refleksije su neizbježne. Očitavanje neželjene refleksije se uvijek nalazi iza objekta koji ih uzrokuje (npr. rub bazena, plutača, kabel itd.) a njihov intenzitet može biti i veći i manji od originala.

Na slici 5.2 je prikazan put kojeg napravi zvučni signal u slučaju neželjene refleksije od ruba.

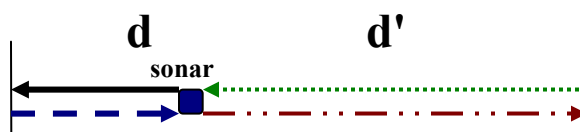
puna crta - zvučni signal emitiran iz sonarske glave

iscrtkana plava – reflektirane signal kojeg sonar zabilježi.

crta dvotočka – zvučni signal se ne zaustavlja kad dođe do ronilice već se i dalje širi dok ne naiđe na prepreku – rub.

točkasta crta – jest neželjeni reflektirani signal

:



Slika 5.2 Neželjena refleksija od ruba

Neželjena refleksija se nalazi na udaljenosti od $d+d'$

- Najveći problem mjerenja su neobjašnjiva uzdignuća relativno velikog intenziteta. Nisu posljedica poznatih smetnji niti ih možemo eliminirati jer njihovo javljanje ne podliježe nekom posebnom pravilu.
- Jedan oblik tih anomalija se javlja kad je ronilica blizu ruba (do cc 0.7 m) i to na udaljenosti oko 2 puta većoj od prave. To je najvjerojatnije refleksija zvučnog signala od ronilice: Odbijeni zvuk dođe do ronilice (vidi sliku 5.3). Ronilica zabilježi signal (udaljenost d), no on ne produži dalje već se odbije od ronilice i ponovi postupak (udaljenost d'). Kako udaljenost određujemo na temelju vremena potrebnog da se odbijeni signal vrati, dobijemo varljive rezultate ($d+d'$). Ove neželjene refleksije su reda veličine prave udaljenosti te ih je teško eliminirati.

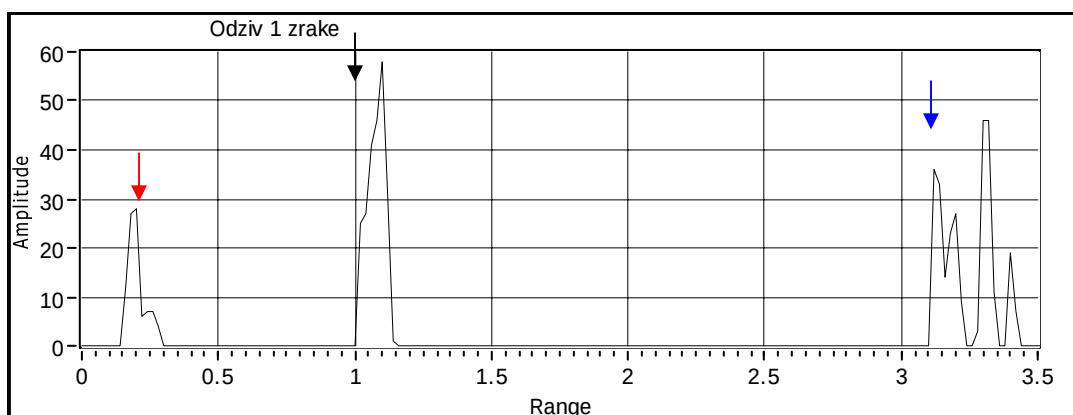


Slika 5.3 Neželjena refleksija od ronilice

Različiti objekti na grafovima skeniranih nizova su označeni različitim strelicama:

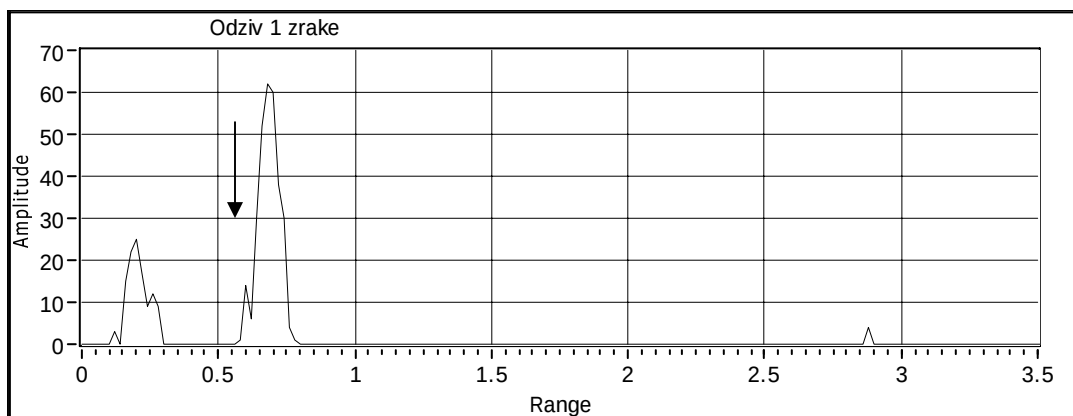
- crna strelica** – rub
- crvena strelica** – šum
- plava strelica** – neželjena refleksija od ruba
- zeleno strelica** – neželjena refleksija od same ronilice
- narančasta strelica** – plutača
- smeđa strelica** – kabel
- siva strelica** – neobjašnjivo uzdignuće

Izražen utjecaj neželjene refleksije od ruba

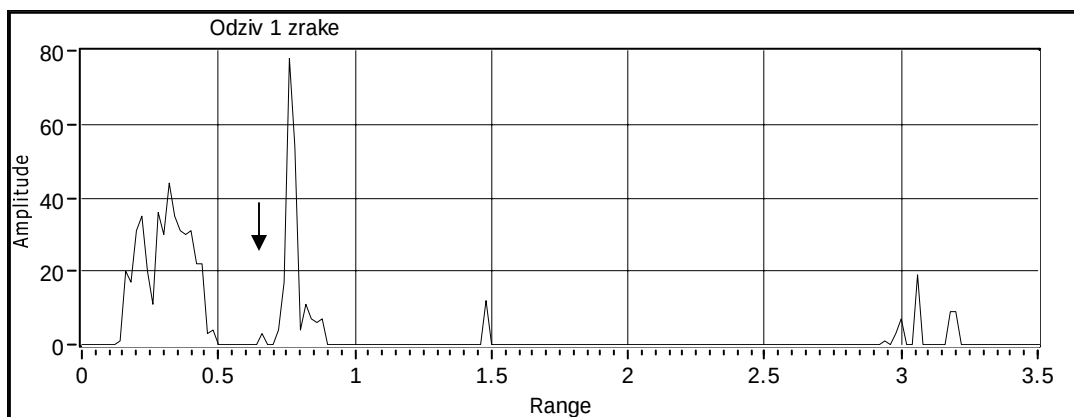


Slika 5.4 Utjecaj neželjene refleksije od ruba

Razlika mjesta maksimalne amplitude i prave udaljenosti

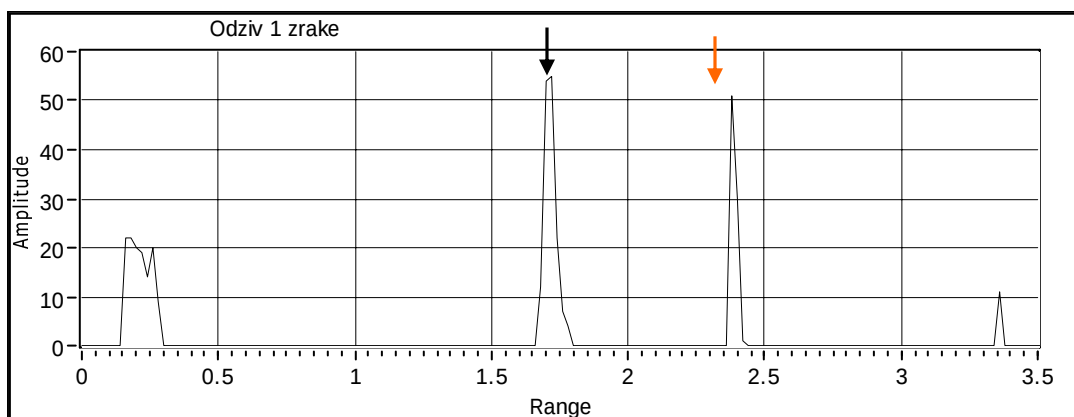


Slika 5.5 Rub ima maksimalnu amplitudu no početak njegova
izdizanja jest nestabilan i puno varira



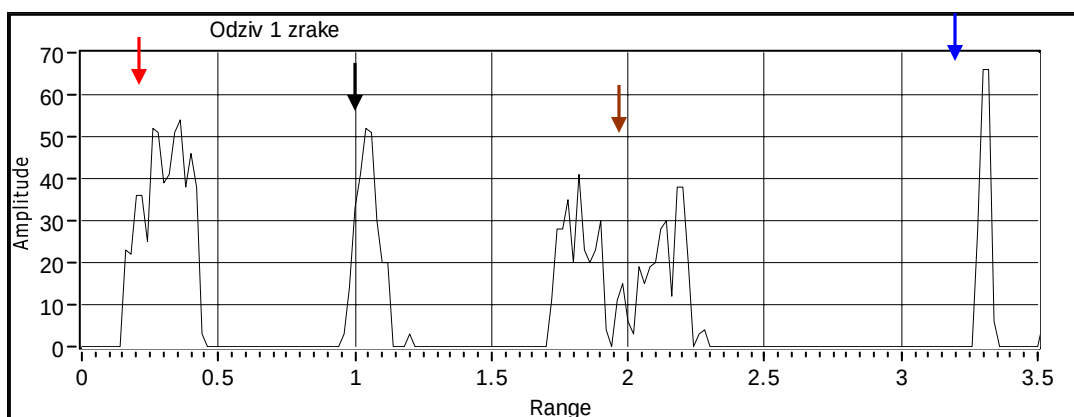
Slika 5.6. Početno izdizanje jest odvojeno od maksimalnog izdizanja

Utjecaj smetnji

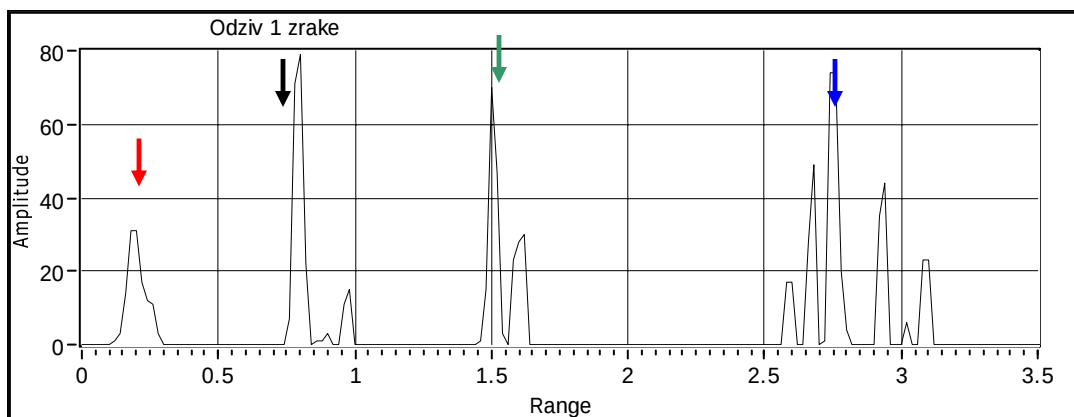


Slika 5.7 Utjecaj plutače : plutača je 1m ispred, a očitana kao 1m iza ruba

Slučaj kad je ronilica malo ispod površine

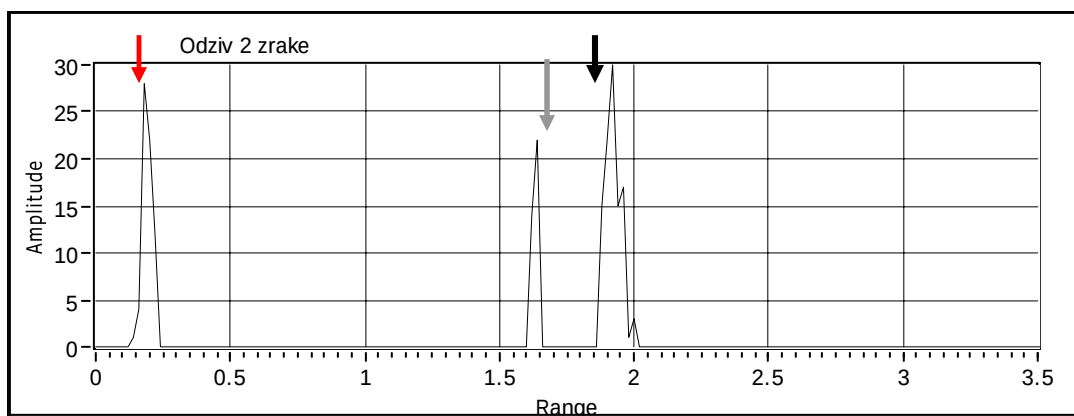


Slika 5.8 Sonar je 20 cm ispod površine. Do izražaja dolazi kabel



Slika 5.9 Sonar se nalazi 10 cm ispod površine. Do izražaja dolazi neželjena refleksija od ronilice

Neobjašnjiva izdignuća

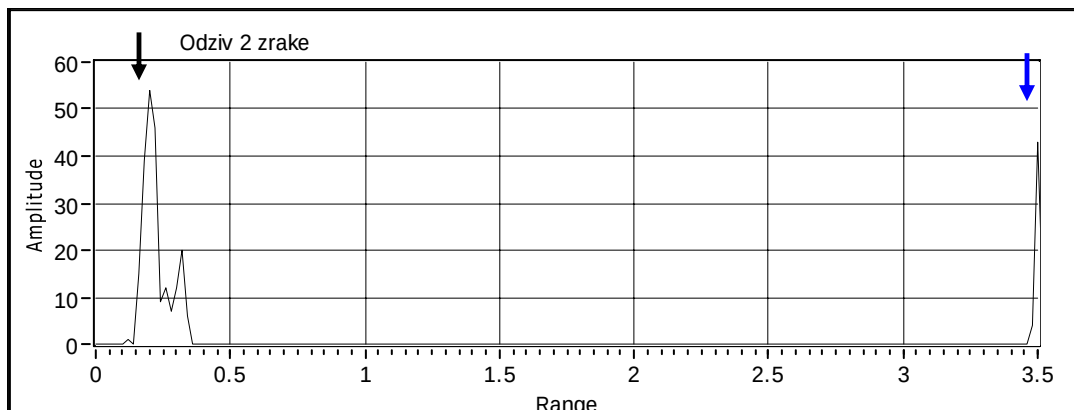


Slika 5.10 Neobjašnjivo izdignuće na 1.6m

Slučaj kad je ronilica blizu ruba

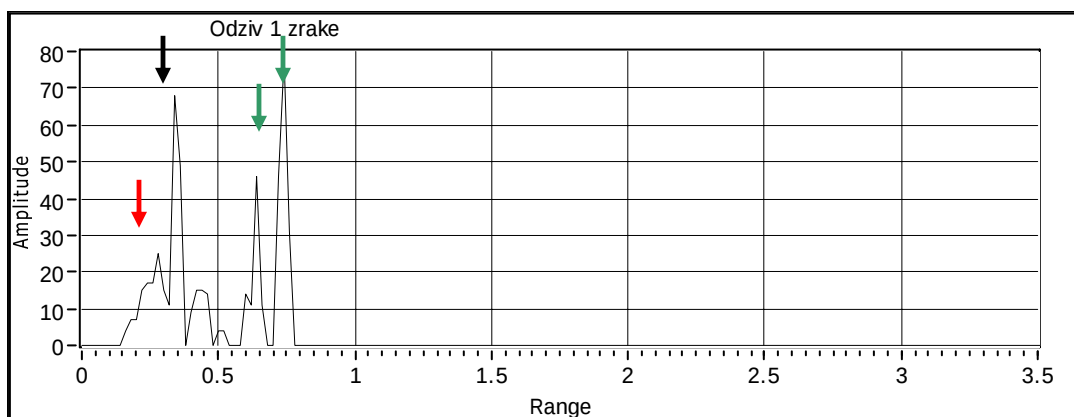
Sonar je neosjetljiv ispod cc 10 cm. Moguće dvije vrste refleksija:

Neželjena refleksija od suprotnog ruba

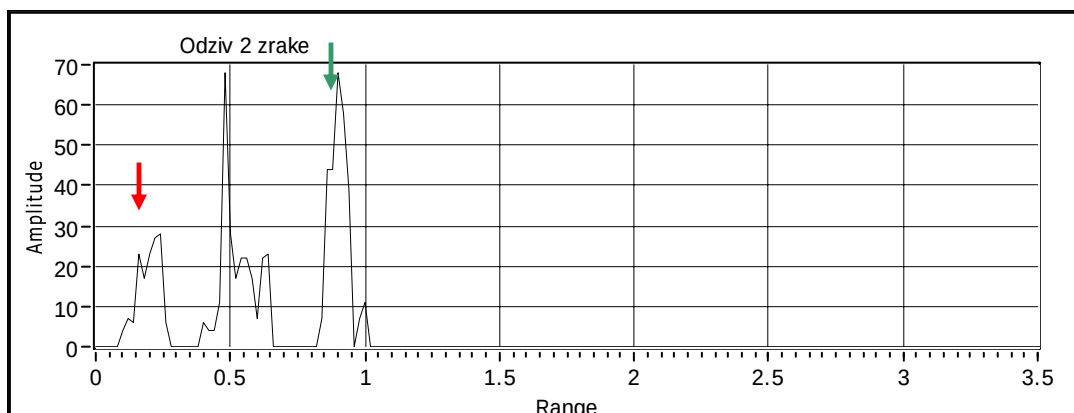


Slika 5.11 Refleksija od ruba kad je ronilica jako blizu ruba

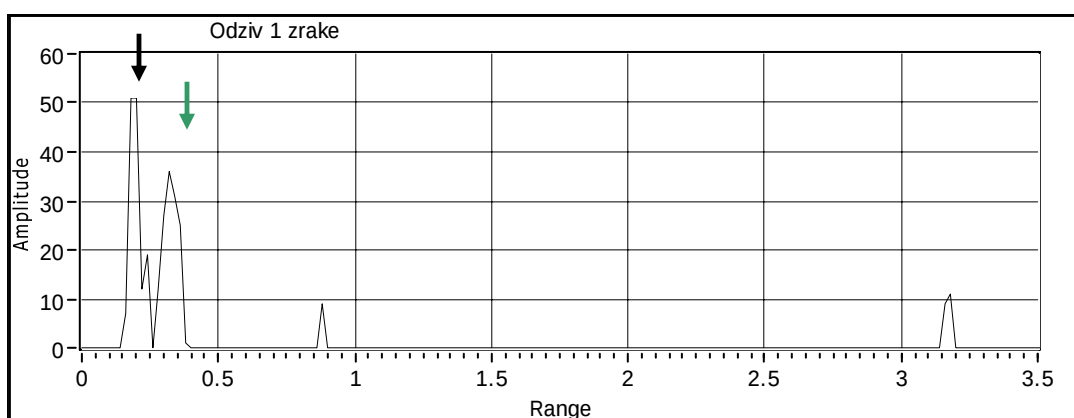
Neželjene refleksije od same ronilice kad je u blizini ruba



Slika 5.12 Neželjena refleksija se nalazi od 0.6m. Moguće ju je eliminirati tako da se za udaljenost uzme prva vrijednost veća od šuma (>30) no to vrijedi samo za dno



Slika 5.13 Još jedan primjer neželjene refleksije od ronilice


 Slika 5.14. Intenzitet neželjene refleksije se smanjuje približavanjem rubu,
a više nema ni šuma jer je on puno manji od intenziteta ruba

Iz snimljenog je uočljivo da postoje situacije gdje je iz skeniranog niza nemoguće odrediti pravu vrijednost udaljenosti. Jedan od mogućih rješenja jest pamćenje nekoliko udaljenosti iz istog prostora skeniranja te njihovom usporedbom pomoću Kalmanovog ili nekog drugog filtra odrediti pravu udaljenost s većom sigurnošću.

5.2.5 Izmjerena preciznost

Mjerenje je izvedeno s krojačkim metrom zalijepljenim na plastični štap (vidi sliku 5.16). Mjera je uronjena u vodu. Zbog oblina ruba bazena, štap ne prijanja uz rub već je udaljen 1.5 cm. U aplikaciju za mjerenje duljine bio je unesen *offset* od 11 cm



Slika 5.15 Ronilica u trenutku mjerenja preciznosti

Prva mjerenja izvršena su na sjevernoj strani bazena (vidi tablicu 5.1).

Tablica 5.1 Rezultati prvog mjerenja

n	d	d'	d-d'	de	course
1.	0.32	0.31	0.01	0.02	324
2.	0.42	0.39	0.03	0	324
3.	0.52	0.49	0.03	0	324
4.	0.61	0.59	0.02	0.01	336
5.	0.72	0.69	0.03	0	334
6.	0.79	0.75	0.03	0	338
7.	0.91	0.89	0.02	0.01	340
8.	1.02	0.97	0.05	-0.02	344
9.	1.11	1.07	0.04	-0.01	344
10.	1.19	1.17	0.02	0.01	346
11.	1.31	1.29	0.02	0.01	335
12.	1.39	1.35	0.06	-0.03	335
13.	1.42	1.39	0.03	0	335
14.	1.47	1.43	0.04	-0.01	336

Greška iznosi: $e = 0.03 \pm 0.0037[m]$.

Druga mjerenja izvedena su na istočnoj strani. Postupak mjerenje je poboljšao uranjanjem štapa u vodu i njegovim usmjeravanjem prema mjeri kako bi izbjegao refleksiju.

Tablica 5.2 Rezultati drugog mjerenja

n	d	d'	e	de	course
1.	0.06	0.15	-0.09	/	224
2.	0.14	0.16	-0.02	-0.023	266
3.	0.23	0.23	0	-0.003	234
4.	0.33	0.33	0	-0.003	236
5.	0.45	0.45	0	-0.003	240
6.	0.53	0.53	0	-0.003	244
7.	0.62	0.61	0.01	-0.002	244
8.	0.73	0.73	0	-0.003	250
9.	0.84	0.85	-0.01	-0.004	252
10.	0.9	0.89	0.01	-0.002	254
11.	1.02	1.01	0.01	-0.002	258
12.	1.09	1.11	-0.02	-0.005	252
13.	1.20	1.21	-0.01	-0.004	262
14.	1.29	1.31	-0.02	-0.005	260
15.	1.41	1.41	0	-0.003	260
16.	1.50	1.51	-0.01	-0.004	262

Greška iznosi: $e = 0.003 \pm 0.0017[m]$

Iz mjerenja se može zaključiti da ofset treba postaviti na vrijednost od 0.1m
 Za maksimalni broj uzoraka NBins=800 po dokumentaciji rezolucija bi trebala biti 7.5 mm, no kako koristim rezoluciju od NBins=200 rezolucija automatski smanjena 4 puta pa teoretski iznosi oko 3 cm. To potvrđuju i prva i druga mjerenja , gdje rezolucija iznosi oko 2-3 cm. Pogreška u drugom mjerenju ne daje dobru sliku o tom zbog velikog broja preciznih mjerenja.

5.2.6 Testiranje lokalizacije u stvarnom vremenu (*Online* Lokalizacija)

Online Lokalizacija koristi aplikaciji koja skenira. Problem je u tome što se ta aplikacija temelji na pretpostavci da sonar miruje dok se ne očitaju udaljenosti. Ako druga udaljenost nije izmjerena iz iste točke kao i prva implementirano geometrijsko rješenje više ne vrijedi. Treba pronaći novo na temelju praćenja trajektorije, no to je tema za jedan novi diplomski rad.

Usporavanjem gibanja ronilice samo sam smanjio neizbježnu grešku. Veličina greške ovisi o položaju ronilice, brzini gibanja i smjeru gibanja. Greške su nedopustivo velike u slučaju da se ronilica:

- nalazi blizu ruba, te je usmjereno giba prema ili uz ruba bazena,
- okreće
- brže giba

Zajednička karakteristika ovih uvjeta jest velika izmjerene i tražene tj. pretpostavljene udaljenosti. I dalje ne postizemo uvijek željeni učinak. Razlog tomu je vrlo mali kut (25°) između udaljenosti, pa točnost više utječe na rezultate algoritma.

Najbolje rezultate dobivamo sporim gibanjem od ruba prema središtu bazena, i obrnuto.

Važno se sjetiti da ni pri mirovanju nemamo uvijek točne rezultate zbog nesavršenog određivanja udaljenosti.

Još jedan nedostatak jest variranje kursa $\pm 2^\circ$ iz čega slijedi i variranje kuta pozicije za $\pm 2^\circ$.

6 ČESTO POSTAVLJANA PITANJA (FAQ)

- 1 Zašto aplikacija za određivanje pozicije nije realizirana kao dio PC pilot-a već kao zaseban VI objekt.**

U programskom paketu LabVIEW iz nepoznatih razloga nemoguće je istovremeno koristiti dva porta koja rade različitim brzinama prijenosa. Support naime ne odgovara na desetke poruka.

- 2 Zašto je potrebno prvo pokrenuti PC pilot a tek onda aplikaciju za određivanje pozicije**

PC pilot i aplikacija međusobno su povezani globalnim varijablama: PC pilot piše kurs na neku memoriju, a aplikacija čita. U slučaju da pokrenemo PC pilot dok aplikacija radi on neće moći odmah zapisati kurs već će morati usporiti petlju. To će poremetiti postupak traženja i odabir porta te komunikacija s ronilicom neće biti spojena.

- 3 Zašto SeaNet Setup program ne želi izvršiti reprogramiranje ‘firmware’-a u sonaru.**

Vjerovatno tipka OK ili Program FPGA nije dovoljno dugo držana pritisnutom.

7 ZAKLJUČAK

Eksperimentalni testovi razvijenih algoritama i aplikacija su pokazali da je lokalizacija u bazenu pomoću dvije zrake i kursa moguća i vrlo precizna. Utjecaj neželjenih refleksija i smetnji je znatan, no ipak u većini slučajeva je moguća i eliminacija, a pozicija je vrlo precizna: Pri mirovanju ronilice pogreška iznosi oko 3 cm i 2° u slučaju dobrog odabira udaljenosti.

Slaba karika u projektiranom algoritma je velika učestalost pogrešnog odabira udaljenosti iz skeniranog niza. Jedno od mogućih rješenja jest da se pri skeniranju prostora osim onih dviju udaljenosti pri krajnjim kutevima, uzmu u obzir i sve ostale izmjerene udaljenosti. Tada bi imali više potencijalnih udaljenosti iz kojih bi se dalo izračunati mjera sigurnosti, potrebna za točniji odabir duljine iz skeniranih nizova na graničnim kutevima.

U budućnosti je potrebno poboljšati i *online* lokalizaciju: Nužno je riješiti geometrijski problem bez pretpostavke da ronilica miruje dok skenira. Dvije udaljenosti ne bi više imale polazište u istoj točki. To zahtjeva složeni matematički proračun jer treba pronaći vezu između razlike udaljenosti na susjednim kutevima i pomaka ronilice.

Ispitivanja su pokazala da je preciznost određivanja udaljenosti reda veličine 10 mm, što garantira i precizan pomak. Povećavanje brzine određivanja pozicije na štetu kvalitete skeniranog niza (npr. manji broj uzoraka) nema prevelikog smisla, pa tu opciju bi trebalo izbjegavati.

Postojeća aplikacija uzrokuje sistemsko kašnjenje od oko 230 ms. Postavlja se pitanje koliko bi to kašnjenje iznosilo u nekim kodnim programima, npr. C++-u, Javi itd.

Zbog loše podrške LabVIEW-a nije pronađen odgovor zašto je onemogućen istovremeni rad dva porta različitim brzinama. Problem je riješen razdvajanjem u više aplikacija, a daljnji cilj je instaliranje novog LabVIEW 7.8.

8 LITERATURA

- [1] Software Notes for controlling and operating RS-232 Sonar Heads : SeaKing / SeaPrince / MiniKing / Micron (Tritech international)
- [2] Tritech Micron OEM Sonar Installation and Operators manual
- [3] Micron DST Startup.PDF
- [4] Službene stranice Tritech international: <http://www.tritech.co.uk/>
- [5] Oliver Svirčić Antišin: Primjena programskog paketa LabVIEW za bespilotnu ronilicu VideoRay Pro II, Diplomski rad br. 1469, Zagreb studeni 2005.
- [6] Z. Kovačić, S. Bogdan, V. Krajči: Osove Robotike, Graphis, Zagreb 2002.
- [7] K. K. Yoong and P. McKerrow : Face Recognition with CTFM Sonar, School of information Technology and Computer Science, University of Wollongong, Australia
- [8] Modern Physics: Math: Vectors, from Wiki Books:
http://en.wikibooks.org/wiki/Modern_Physics:Math:Vectors
- [9] Derivation of rotation matrix using polar coordinates, from PlanetMath.org:
<http://planetmath.org/?op=getobj&from=objects&name=DerivationOfRotationMatrixUsingPolarCoordinates>
- [10] LabVIEW : Serial Communication Overview, s interneta
<http://zone.ni.com/devzone/conceptd.nsf/webmain/8DECBF3E0B714BF3862568F9006E7851>

9 DODATAK A – algoritam za određivanje udaljenosti

```

int32 i,pom,i_max,poc,kraj,ni;
float d,d1,d1_p,noise,tol,ad1,ad1_p;
//=====racunam shum , i d pod pretpostvkom da nema refleksije=====//
pom=0;
noise=0;
//dk=dk-0.15;           // reducirali promjer za pol. dulj ron
for(i=0;i*dx<0.3;i++) { //pronalazim shum
    if (y[i]>noise) {
        noise=y[i],ni=i;
    }
}

for(i*dx<dk;i++){        //trazenje maksimuma
    if (y[i]>pom){
        pom=y[i]; i_max=i; ad1=y[i];
    }
}
d1=i_max*dx;
d=d1;
//=====racunam d pod pretpostavkom da je nema smetnji i da je sum puno manji =====//

pom=0; i_max=0; tol=0;
for(i=0;i*dx<0.3;i++)           //pozicioniram na podrucje osjetljivosti
    tol=0.2*noise;
for(;y[i]>tol;i++);               //trazim prazninu
for(;y[i]<=tol;i++){              //trazim poc uzorkovanja udaljenost uzavajuci toleranciju u slucaju da naidje plutaca
    if(i*dx>dk) break;
}

poc=i;
for(;y[i]>0;i++) {                //trazim kraj uzorkovanja
    if (i*dx>dk) break;
}
kraj=i;                          //trazim max udaljenost
for(i=poc;i<kraj;i++){
    if (y[i]>pom){
        pom=y[i]; i_max=i; ad1_p=y[i];
    }
}
d1_p=i_max*dx;
//===== slijedi odlucivanje koji d je pravi d =====//
if (abs(ad1-ad1_p)<ad1*0.3) d=d1_p;

if (ad1< noise){                 //usporedba shuma i max d (d1)

    if (d>dk*0.7) d=ni*dx; // zanemariti jako daleku refleksiju samog suma
    if (noise-ad1>noise*0.5) d=ni*dx; //sum mora bit puno veci da bi bio vrijednost
}
d=d- offset;                    // uracunavam ofset

```

10 DODATAK B – algoritam za određivanje pozicije

U ovom algoritmu varijable nisu identične onima u objašnjenju u poglavlju 4.5. Jedna od bitnih razlika jest da je u kodu radijus označen s r , a udaljenost od središta s x .

```
float std,fi,x,a,kapa,epsilon,fi_prva,fi_druga,beta,ni,alfa,d2_prva,d2_druga,axis;
delta=delta*pi/180;
psi=psi*pi/180;
//racunanje x-a
a=sqrt(pow(d1,2)+pow(d2,2)-2*d1*d2*cos(delta));
kapa=acos(pow(a,2)/(2*a*r));
epsilon=acos((pow(a,2)+pow(d2,2)-pow(d1,2))/(2*a*d2));
alfa=kapa-epsilon;
x=sqrt(pow(r,2)+pow(d2,2)-2*r*d2*cos(alfa));
//racunanje fi-a

std=acos((pow(d1,2)+pow(x,2)-pow(r,2))/(2*d1*x));
axis=((pow(d1,2)+pow(x,2)-pow(r,2))/(2*d1*x));
//slijedi racunanje fi-a
//kako ne mogu znati na kojoj se polutki nalazimo izracunat
//cu fi za obe polutek tj. fi_pva i fi_druga
fi_prva=pi+psi-std;
fi_druga=-pi+psi+std;

//treba izracunati izsimulirani d2 za fi_prva i fi_druga

//slijedi racunanje druge udaljenosti za fi_prva

beta=pi-(psi+delta-fi_prva);
ni=asin(x*sin(beta)/r);
alfa=pi-beta-ni;
d2_prva=sqrt(pow(r,2)+pow(x,2)-2*r*x*cos(alfa));

// racunanje druge udaljenosti za fi_druga

beta=pi-(psi+delta-fi_druga);
ni=asin(x*sin(beta)/r);
alfa=pi-beta-ni;
d2_druga=sqrt(pow(r,2)+pow(x,2)-2*r*x*cos(alfa));

// usporedba d2_prva i d2_druga
if(abs(d2_prva-d2)<abs(d2_druga-d2)) fi=fi_prva;
else fi=fi_druga;

// specials situations
if (d1==r && d2==r) fi=psi;

//if (d1==0) ova situacija nemoguca
// if 115<delta<285 kriticno ne racuna dobro
fi=fi*180/pi;
fi_prva=fi_prva*180/pi;
```

```
fi_druga=fi_druga*180/pi;
```

```
if (fi<0) fi=360+fi;
```

```
if (fi>360) fi=fi-360;
```

11 DODATAK C – algoritam za rotaciju i translaciju

```
float r_t,fi_t,xk,yk,xkr,ykr,xkt,ykt,xk0,yk0;
```

```
r_k=r_k*radius/1.75;
```

```
fi0=fi0*pi/180;
```

```
fi_k=fi_k*pi/180;
```

```
psi=psi*pi/180;
```

```
xk=r_k*cos(fi_k);
```

```
yk=r_k*sin(fi_k);
```

```
xkr=xk*cos(psi)-yk*sin(psi);
```

```
ykr=xk*sin(psi)+yk*cos(psi);
```

```
xk0=r0*cos(fi0);
```

```
yk0=r0*sin(fi0);
```

```
xkt=xkr+xk0;
```

```
ykt=ykr+yk0;
```

```
r_t=sqrt(pow(xkt,2)+pow(ykt,2));
```

```
fi_t=atan2(ykt,xkt);
```

```
if (xkt==0) fi_t=0;
```

```
fi_t=fi_t*180/pi;
```

12 DODATAK D – algoritam za pozicioniranje u pravokutniku

```

int yq,cas,ang_x,ang_y,b,ca_m,angx_MP;
yq=0;
//podesavanje u int [0-180] i [0,-180]
if (ca>360)
ca=ca-(floor(ca/360))*360;
if (ca>180)
ca=ca-360;
if (ca<-360)
ca=ca-(floor(ca/360))*360;
if (ca<-180)
ca=ca+360;
ca_m=ca;
////////////////////
if (ca>=0) {
    if (ca<90){
        cas=1; // prvi kvadrant
        ang_x=ca;
        ang_y=90+ca;
    }
}
if (ca>=90) {
    if (ca<=180){
        cas=2; // drugi kvadrant
        ang_x=ca;
        ang_y=ca-90;
    }
}
if (ca<=-90) {
    if (ca>=-180){
        cas=3; // treci kvadrant
        ang_x=180+ca;
        ang_y=270+ca;
    }
}
if (ca<0) {
    if (ca>-90){
        cas=4; // cetvrtii kvadrant
        ang_x=180+ca;
        ang_y=90+ca;
    }
}
/////pretvorba kuta ang x iy u MP//

```

```
ang_x=(ceil(ang_x/1.8)*32)+MPLeft;  
ang_y=(ceil(ang_y/1.8)*32)+MPLeft;  
////////za slucaje MP=6400////////  
if (ang_x==6400) ang_x=0;  
if (ang_y==6400) ang_y=0;  
////////trazenje MP-a/////////  
if (x==ang_x) yq=1;  
if (x==ang_y) yq=3;
```

Sažetak

U diplomskom radu je primjenom programskog paketa LabVIEW realizirana serijska komunikacija s sonarom Tritech Micron. Sonar je kompatibilan za rad s ronilicom VideoRay Pro II. Razvijeno je sučelje pomoću kojeg će operator moći odrediti dvodimenzionalnu poziciju ronilice u strukturiranoj okolini kao što su bazeni, razervoari, vodotornjevi i slično. Izvršena je usporedba preciznosti i brzine kontinuiranog rada i rada sonara s dvije zrake. Eksperimentalno su ispitane mogućnosti sonara u labaratoriskom bazenu te obavljena analiza dobivenih rezultata.

Ključne riječi: Ronilica, Video Ray Pro II, Tritech Micron, sonar, LabVIEW, pozicioniranje, lokalizacija, kodiranje, dekodiranje, skenirani niz.

Abstract

Using LabVIEW platform is developed communication over serial port (RS232) with sea sonar Triton Micron. This sonar is originally compatible with remote operated vehicle (ROV) VideoRay Pro II. It has been designed an interface for two dimension vehicle positioning in known environment. like pools, tanks, water towers etc.appropriate. Various modes of sonar scanning are tested in circle pool and the results are analysed.

Key words: ROV, VideoRay Pro II, Triton Micron, sonar, LabVIEW, positioning, localization, encoding, decoding, scanline.

Životopis

Zdravko Eškinja rođen je u Zadru 3. kolovoza 1983. godine, a do 2001 živi u Biogradu na Moru. Iste godine završava opću gimnaziju u srednjoj školi “Biograd na Moru” i upisuje Fakultet elektrotehnike i računarstva u Zagrebu. Kao student radi na mnogim fakultatetskim projektima i volontira u nezavisim i nepolitičkim udrugama (BEST, Zelena Akcija). Poznavanje Engleskog jezika jest odlično, dok njemački i talijanski koristi samo u svrhu najosnovnijeg sporazumijevanja.

Pod vodstvom mentora prof. dr. sc. Z. Vukića proučavao mogućnosti i iskoristivost Tritech Micron sonara u svrhu online ronilice VideoRay Pro II što je i tema njegovog diplomskog rada.

Potpis _____