Залуурдагддаггүй тэнхлэг бүхий роботын координатыг хугацааны мэдээлэл ашиглаж олох алгоритм

Г.Ганбат, Д.Нанзадрагчаа, М.Баярпүрэв
Монгол Улсын Их Сургуулийн
Мэдээллийн Технологийн Сургууль
Улаанбаатар хот, Монгол Улс
Email: www.ganbat_net@yahoo.com

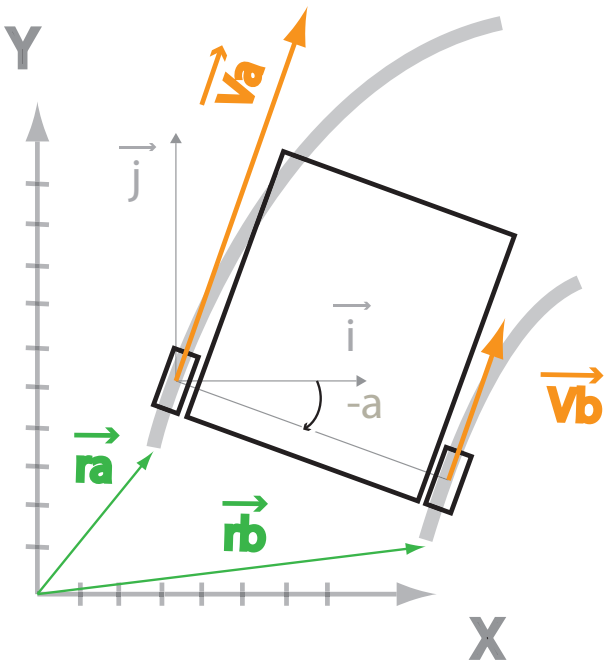
Хураангуй—Роботын координатыг өндөр нарийвчлалтайгаар, real-time олох бодлого нь роботикийн чухал асуудал билээ. Энэхүү судалгааний ажилд роботын залуурдагддаггүй тэнхлэг дээр байрлуулсан ротари энкодер ашиглан координатыг тооцоолох алгоритмыг санал болгоно. Эргэлдэгч энкодерийн тасалдал үүсэх хугацааг бүртгэж тухайн цэгийн хурдны фүнкцийг ойролцоолон сэргээнэ. Улмаар үл залуурдагдах тэнхлэг дээрх цэгийн хурд тэнхлэгтэй перпендикуляр байдаг гэдгийг ашиглан кинематикийн тооцоолол хийнэ. Алгоритмыг OMAP4430 процессор бүхий Panda board ашиглаж туршилтыг хийсэн.

Түлхүүр үгс: Роботикс, координат олох, машин-хэвт робот, зам төлөвлөлт

I. Оршил

Сүүлийн жилүүдэд жолоочгүй машины судалгаа хөгжүүлэлт эрчимтэй явагдаж байна. Америкийн батлан хамгаалах яам ч энэ салбарыг анхаарч жил бүр DARPA Grand Challenge тэмцээнд жолоочгүй машины уралдаан явуулдаг. Түүнчлэн, үйлдвэрлэгч нар ч энэ салбарт өрсөлдөж эхлээд байна. Гүүгл компаний жолоочгүй машин, Тоётагийн i-unit концепт машин зэрэгийг дурдаж болно. Жолоочгүй машин эсвэл робот нь өөрийн координатыг маш нарийн мэдэж байх шаардлага гардаг. Үүнд GPS хиймэл дагуулыг ашиглаж болох хэдий ч нарийвчлал маш муу болно (Differential GPS ашигласан тохиолдолд 1м, бусад үед 7м). Радио долгион ашиглан үүнээс нарийн хэмжихийн тулд Ultra-wide-band технологи ашиглаж болох ч дэд бүтэц нь үнэтэй, хэрэглэгдэх хүрээ багатай болчихдог. Камер ашигласан хиймэл оюун ухааныг хэрэглэж болох хэдийг ч энэ нь нүсэр тооцоолол шаардах бөгөөд real-time чанараа алддаг дутагдалтай. Машины үл залуурдагдах тэнхлэг дээр байрлуулсан ротари энкодер ашиглан координат олох алгоритм маань тооцоолол багатай, хямд, real-time зэрэг давуу талуудтай. Энэ аргыг дээрх аргуудтай хавсран ашиглавал илүү үр дүнтэй. Энэ өгүүлэлийг дараах байдлаар зохион байгуулсан болно: II дэд бүлэгт залуурдагддагүй тэнхлэг бүхий роботын кинематик, траекторын анализыг хийнэ. III дэд бүлэгт бидний санал болгож буй ГАЛ алгоритмын талаар тайлбарлах бөгөөд IV дэд бүлэгт туршилтын үр дүн, V дэд бүлэгт дүгнэлтээ танилцуулна .

II. Залуурдагддагүй тэнхлэг бүхий роботын кинематик



Зураг 1: Роботын траектор анализ

Дугуй тус бүрийг харгалзан a,b гэвэл түүний үүсгэх замын илэрхийлэл нь :

$$\vec{r}_a(t) = \begin{pmatrix} x_a(t) \\ y_a(t) \end{pmatrix}, \quad \vec{r}_b(t) = \begin{pmatrix} x_b(t) \\ y_b(t) \end{pmatrix} \tag{1}$$

1 гэе. Эдгээр замын илэрхийлэлээс хурдны вектор тус бүр нь :

$$\vec{v}_a(t) = \vec{r}_a(t)' = \begin{pmatrix} x_a(t)' \\ y_a(t)' \end{pmatrix}, \vec{v}_b(t) = \vec{r}_b(t)' \tag{2}$$

Дугуй тус бүрийн үүсгэж буй траекторийн тухайн агшин дахь ширгэгч шулуун нь тухайн дугуйн хурдны векторын дагуу байна. Цаашлаад, хэрэв хальтралтыг үл тооцвол тухайн дугуйн чиглэл нь ч мөн энэхүү хурдны векторын дагуу чиглэнэ. Иймээс хойд хоёр дугуйн $\vec{v}_a(t)$, $\vec{v}_a(t)$ векторууд хоорондоо параллель бөгөөд AB талтай перпендикуляр чиглэнэ. Тухайлбал, хэрэв роботын A дугуй урагш эргэж байсан гэж үзвэл $\vec{v}_a(t)$ -ийг цагийн зүүний дагуу 90° эргүүлбэл $\overrightarrow{ab}$ векторын дагуу чиглэнэ. Иймээс,

$$\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \frac{1}{|\overrightarrow{AB}|}\overrightarrow{AB} = \frac{1}{|\vec{v}_A(t)|}\begin{pmatrix} 0 & 1 \\ -1 & 1 \end{pmatrix}\vec{v}_A(t) \quad (3)$$

буюу

$$\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \begin{pmatrix} \frac{y_A(t)'}{|\vec{v}_A(t)|} \\ -\frac{x_A(t)'}{|\vec{v}_A(t)|} \end{pmatrix} \quad (4)$$

болно. Түүнчлэн,

$$\vec{r}_B = \vec{r}_A + \overrightarrow{AB} = \vec{r}_A + \frac{L}{|\vec{v}_A|}\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\vec{v}_A \quad (5)$$

болно. Энэд хувсагч t-ийг орхин товчлов. Ро-та өнцөг хурдыг $\omega(t) = \theta(t)'$ гэж тэмдэглэн нормалчлагдсан хурдны вектороос уламжлал авбал

$$\frac{d}{dt}\left(\frac{\vec{v}_A}{|\vec{v}_A|}\right) = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}\frac{\vec{v}_A}{|\vec{v}_A|}, \quad (6)$$

$$\Rightarrow \quad \omega = \left|\frac{d}{dt}\left(\frac{\vec{v}_A}{|\vec{v}_A|}\right)\right| \quad (7)$$

болно. Өөрөөр хэлбэл энэ уламжлал авах үйлдэл нь энэ векторыг ω дахин сунгаж цагийн зүүний эсрэг 90° эргүүлэхтэй адилхан. Эндээс, (2) томъёоны хурдыг дахин тооцолбол

$$\vec{v}_B = \vec{v}_A + \begin{pmatrix} 0 & L \\ -L & 0 \end{pmatrix}\begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}\frac{\vec{v}_A}{|\vec{v}_A|} \quad (8)$$

$$\Rightarrow \quad \vec{v}_B = \vec{v}_A + \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}\overrightarrow{AB} \quad (9)$$

болно. Эдгээр томъёонуудад эргэлтын матрицуудын байр солидог чанарыг ашиглав.

Эндээс v_B -г :

$$v_B = |\vec{v}_B| = |\vec{v}_A| + L\omega \quad (10)$$

болно. Дээрх томъёоноос харахад роботыг өгөгдсөн траекторoор өгөгдсөн хурдтай явуулахын тулд $|\vec{v}_A|(t)$ болон ω-г тооцоолж даалгавар болгоход хангалттай болох нь илэрхий. ω ро-та өнцөг хурд нь зөвхөн A дугуйн траекторoос хамаарах ба дугуйн хурднуудаас хамаарахгүй. Мөн $|\vec{v}_A|(t)$ хурдны векторыг траекторoос хамааралгүйгээр өгч болно. Энэ хоёр хувьсагчаас бусад бүх хувьсагчдыг гарган авч болно. Хурдны векторуудыг $\vec{v}_A(t)$-тай харьцангуй өнцгөөс нь хамааран абсолют утгыг нь сөрөг утга оноож өгч болно:

$$\text{sign}(v_B) = \text{sign}(\vec{v}_A \cdot \vec{v}_B) = \begin{cases} +1, & v_A > \omega L \\ -1, & v_A \le L\omega \end{cases} \quad (11)$$

Өөрөөр хэлбэл $\vec{v}_B(t)$ хурдны векторийн $\vec{v}_A(t)$-тай үүсгэх өнцөг дэлгэмэл бол сөрөг утга авна.

III. Координат тооцоолох алгоритмууд

Бид координат тооцолох гурван алгоритм боловсруулсан. Эдгээрийг ГАЛ, ГАЗ, NF гэж нэрлэсэн. ГАЛ алгоритм нь энкодерын тасалдал болон хугцааны мэдээллийг ашиглаж координатаа олдог.

$$\vec{r}_A(t) = \vec{r}_A(0) + \int_0^t \left\|\frac{d}{dt}\vec{r}_A(\tau)\right\|\begin{pmatrix} -\sin\theta(\tau) \\ \cos\theta(\tau) \end{pmatrix} d\tau \quad (12)$$

$$\Rightarrow \quad \vec{r}_A(0) + \int_0^t \left\|\vec{v}_A(\tau)\right\|\begin{pmatrix} -\sin\theta(\tau) \\ \cos\theta(\tau) \end{pmatrix} \quad (13)$$

ГАЗ нь хурдны векторуудыг Диракийн делта $\delta(t)$ функцээр ойролцоолж эргэлтүүд A, B дугуй дээрх эргэлтүүд болно.

$$\theta(t) = \theta(0) + \frac{1}{L}\Big(\int_0^t \left\|\vec{v}_B(t)\right\|d(\tau) - \int_0^t \left\|\vec{v}_A(t)\right\|d(\tau)\Big) \quad (14)$$

$$\Rightarrow \quad \theta(0) + \frac{1}{L}\big(\int_0^t \vec{v}_B(t)d(\tau) - \int_0^t \vec{v}_A(\tau)d\tau\big) \quad (15)$$

$$\Rightarrow \quad \theta(0) + \frac{1}{L}\big(s_B(t) - s_A(t)\big) \quad (16)$$

Алгоритмуудын алдааг үнэлэхийн тулд тооцонгийн аргаар симмүляц хийсэн. Ингэхийн тулд роботын явсан замыгБизьегийн олон гишүүнтээр өгсөн. P_0, P_1, P_2, P_3 өгөгдөх Бизьегийн олон гишүүнтийг эхлэн бичвэл:

$$\vec{r}_A(t) = \mathbf{A}t^3 + \mathbf{B}t^2 + \mathbf{C}t + \mathbf{D}, \quad \mathbf{A} = \mathbf{P}_3 - 3\mathbf{P}_2 + 3\mathbf{P}_1 - \mathbf{P}_0, \quad (17)$$
$$\mathbf{B} = 3\mathbf{P}_2 - 6\mathbf{P} + 3\mathbf{P}_0, \quad \mathbf{C} = 3\mathbf{P}_1 - 3\mathbf{P}_0, \quad \mathbf{D} = \mathbf{P}_0$$

болно. Энэ сегментэд хурдны векторын абсолют утгыг тооцоолбол болохыг хялбархан харж болно:

$$v_A = \sqrt{9\,|\,\mathbf{A}\,|^2\,t^4 + \mathbf{F}t^3 + \mathbf{G}t^2 + \mathbf{H}t + |\,\mathbf{C}\,|^2}$$
$$\mathbf{F} = 12\mathbf{A}\cdot\ \mathbf{B}, \quad \mathbf{G} = 4\,|\,\mathbf{B}\,|^2 + 6\mathbf{A}\cdot\mathbf{C}, \quad \mathbf{H} = 4\mathbf{B}\cdot\mathbf{C} \quad (18)$$

энэд, $\cdot$ -ээр хоёр векторын скаляр үржвэрыг тэмдэглэв.

Дээрх томъёоноос харахад A дугуйн траекторыг Бизье муруйгаар өгсөн тохиолдолд бусад дугуй траектор нь $1/v_A(t)$ үржэгдэхүүн агуулж байгаа тул олон гишүүнт болж чадахгүй нь илэрхий. Траекторыг байгуулдаг программыг Java хэл интерактив болон визуал байдлаар бичиж, дэлгэрэнгүй анализыг тооцонгийн аргаар Matlab программ дээр гүйцэтгэсэн.

A. Траекторыг Тэнцүү Урттай Хэсгүүдэд Хуваах Алгоритмууд

Роботод учруулах тооцооллын ачааллыг бууруулахын тулд тогтмол урттай зам туулсны дараагаар дараагийн даалгарыг өгөх шаардлага гарсан. Үүнийг хэрэгжүүлэхийн тулд аль нэгэн дугуйн траекторыг тэнцүү хэсгүүдэд хуваах алгоритм боловсруулах хэрэгцээ гарсан. Энэ бодлогын тавилтыг доорх байдлаар томъёолж болно:

Хавтгайд $\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D} \in \mathbb{R}^2$ векторууд өгөгджээ. Тэдгээрээр байгуулагдсан $\vec{r}_A(t) = \mathbf{A}t^3 + \mathbf{B}t^2 + \mathbf{C}t + \mathbf{D}$, $t \in [0,1]$ муруйг өгөгдсөн $\Delta \in \mathbb{R}$ урттай хэсгүүдэд хуваа. Тодруулбал, хуваалтын $t_0 = 0, t_1, \cdots, t_K$ агшиныг ол.

$\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}$ векторуудын талаар дэлгэрэнгүйг ?? дэд бүлгээс үзнэ үү. Хувьсагч t нь бодит хугацааг илэрхийлэхгүй бөгөөд Бизьегийн муруйн параметр төдий юм. Жинхэнэ бодит хугацааг болон хурдыг траекторын тэгшитгэлээс хамааралгүйгээр өгнө.

Бид хоёр алгоритм боловсруулж үзсэн. Эхний алгоримт нь математик тооцолол байгаатай бөгөөд Java, Visual Studio зэрэг ердийн программчиллын орчинд гүйцэтгэх боломжтой. Нэгдүгээр алгоритмын санаа энгийн бөгөөд хуваалтын агшныг рекурсив байдлаар тооцоолно:

$$t_0 = 0,\ t_k = t_{k-1} + \frac{\Delta}{v_A(t_{k-1})},\ k = 1,2,\cdots \tag{19}$$

энэд, $v_A(t_{k-1})$ нь хурдны векторын абсолют утга ба (18) томъёогоор өгөгднө. Нэгдүгээр алгоритм нь математик тооцоолол багатай ч хэсгүүдийн урт Δ ихэсвэл нарийвчлал нь муудах дутагдалтай.

Хоёрдугаар алгоритм нь математик тооцоолол ихтэй боловч нарийвчлал өндөртэй. Энэ алгоримт нь өгөгдсөн Бизьегийн муруйн натурал параметрыг олж түүнийг Δ урттай хэсгүүдэд хуваах байдлаар хугацааны агшинг олно. Натураль параметр гэдэг муруйн нумын урт s-ээс хамаарсан хугацааны функц $t = t(s)$ юм. Энэ тохиолдолд $v_A(t(s))$ хурдны абсолют утга нь s параметрын хувьд 1-тэй тэнцнэ гэсэн үг юм:

$$s(t) = \int_0^t v_A(\tau)d\tau, \tag{20}$$

$$v_A(t(s)) = \left\| \frac{d\vec{r}_A(t(s))}{ds} \right\| = \left\| \frac{d\vec{r}_A(t(s))}{dt}\frac{dt}{ds} \right\| = \left\| \frac{d\vec{r}_A(t(s))}{dt} \right\| \frac{dt}{ds} = v_A(t)\frac{dt}{ds} = 1 \tag{21}$$

Натурал параметрыг олохын тулд (21) дифференциал тэгшитгэлийг бодно. Энэ тэгшитгэлд $v_A(t)$-г (18) томъёоноос орлуулан тавибал дифференциал тэгшитгэлийн гарна:

$$t' = \frac{1}{v_A(t)} = \frac{1}{\sqrt{9 \mid \mathbf{A} \mid^2 t^4 + \mathbf{F}t^3 + \mathbf{G}t^2 + \mathbf{H}t + \mid \mathbf{C} \mid^2}} \tag{22}$$

Энэ шугаман бус дифференциал тэгшитгэлийн шийд нь гипергеометр функцээр илэрхийлж болох боловч бидний зорилго натурал параметрыг болох бус түүний Δ урттай хэсгүүд дээрх утга л хэрэгтэй тул тэдгээрийг тооцонгий аргаар бодов. Тодруулбал, $s = k\Delta,\ k = 0,1,\cdots,K$ дээрх $t(n\Delta),\ k = 0,1,\cdots,K$ утгыг Matlab-ын ode45 түүлээр тооцоолов. Эдгээр хуваалтын утгуудыг туршилтад ашиглав.

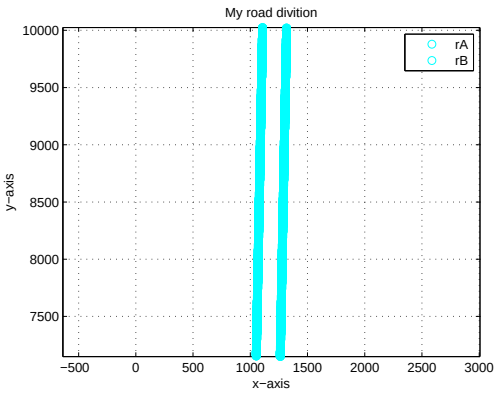
IV. Туршилт

Траекторыг өмнөх хэсэгт заасан дагуу тэнцүү хэсэгт хуваасан. Хуваалт бүрд харгалзах муруйн t параметртыг ашиглан $\vec{r}_A(t)$ болон $\vec{r}_B(t)$ векторуудын координатыг хоорондон эрэмблэж дараалал үүсгэсэн. Энэхүү дараалал нь Залуурдагддаггүй тэнхлэг бүхий роботын ротари энкодерын үүсгэх тасалдалын дараалалтай жишиж болохуйц мэдээлэл юм. Өөрөөр хэлбэл даалгавар траектор нь роботын туулсан зам мэтээр загварчлаж болох юм аа. Даалгавар траекторыг шулуун,дан нумаас бүрдсэн,өлөн нум агуулсан байдлаар дүрсэлж туршиж үзсэн. Энэхүү туршилтаар ГАЛ алгоритмыг бусад ГАЗ, NF зэрэг алгоритмтай жишиж алдааг үнлэж харицуулалт хийсэн.Даалгавар траекторыг байгуулахдаа ″ABU Robocon 2013″ тэмцээний тоглолтын талбайн зургыг суурь зураг болгон ашиглаж байгаа. Хэмжээний хувьд 10:1 харьцаанд оруулсан болно.

Суурь координат дээр даалгавар траекторын дүрслэлийг бизьегийн муруйн куб эрэмбийн тэгшитгэл буюу 4 цэгээр 1 муруйг дүрслэж байгаа. Энэхүү 4 цэгийн мэдээлэлийг ашиглаж буцааж муруйг байгуулах бөгөөд энэ манай роботын A дугуйн туулах траектор болгон ашиглах юм аа. Роботын траектор илэрхийлэл болох (1) болон (5) тэгшитгэлийг ашиглаж бусад траекторыг байгуулдаг.

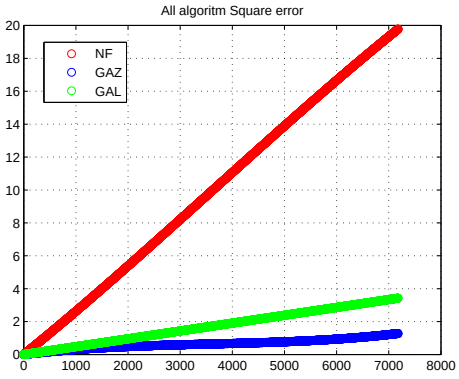
A. Шулуун траектор

Зураг (2) дээр шулуун траекторыг хэрхэн байгуулсаныг харж болно. Роботын нэгж алхамын

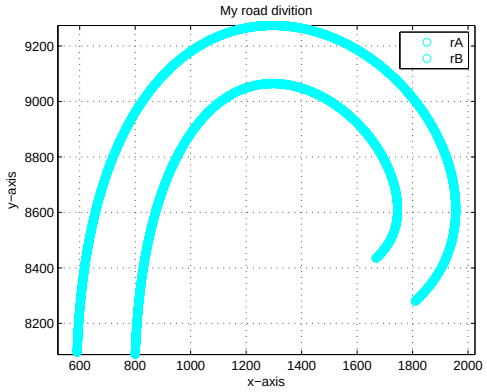


Зураг 2: Шулуун траекторын даалгавар

хэмжээ буюу Δ хэмжээ:
1. ротари энкодерын хуваалтын хэмжээ
2. энкодерт ашиглаж буй дугуйн радиусаас хамаардаг.
Зураг (3) дээр $\Delta = 0.4mm$ үед координат олох алгоритмуудын алдааг харьцуулан харуулсан байна. GAZ алгоритмын алдаа бусад алгоритмтай харьцуулхад бага байгаа нь алгоритмын бодолтоос хамаарч байгаа. Энкодерын тасалдалын утга шулуун явах үед нэг нэгээр дараалаж үүсэх нь их бөгөөд энэ нь алгоритмын координат олох дараалалтай адилхан байгаатай холбоотой.
Зураг (4) дээр $\Delta = 0.8mm$ үед алгоритмуудын алдаа хэрхэн өөрчлөгдөснийг харуулж байна. Нийт алдаа нь ерөнхийдөө их болсон. Өөрөөр хэлбэл Δ хэмжээ нь

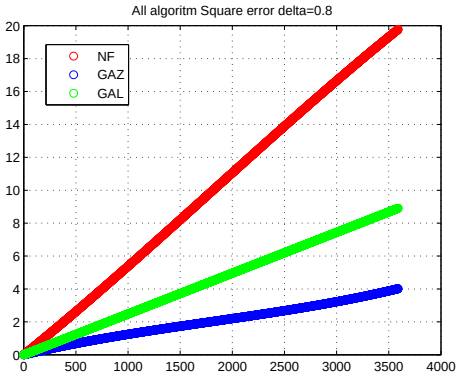


Зураг 3: Алдааны харьцуулалт, $\Delta = 0.4mm$

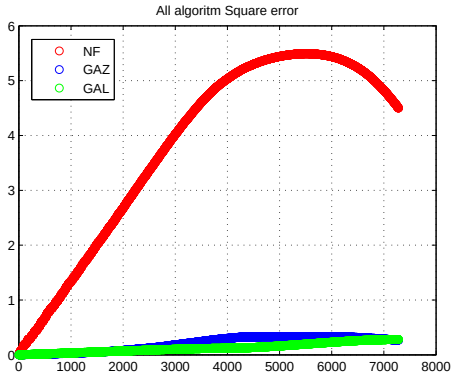


Зураг 5: Нэг муруйгаар илэрхийлсэн даалгавар траектор

бага байхад аль ч алгоритмын хувьд алдаа нь багсах нь харагдаж байна.



Зураг 4: Алдааны харьцуулалт, $\Delta = 0.8mm$



Зураг 6: Алдааны харьцуулалт, $\Delta = 0.4mm$

B. Нэг муруй бүхий траектор

Нэг муруйгаар илэрхийлсэн нум хэлбэртэй траектор даалгаварыг сонгон авсан. Зураг (5) дээр дугуйн координатууд хэрхэн байгуулагдсаныг харж болно. Зураг (6) дээр $\Delta = 0.4mm$ үед координат олох алгоритмуудын алдааг харьцуулан харуулсан байна. Энэ үед GAZ алгоритмын алдаа ГАЛ алгоритмын алдааны утгыг бодвол бага байгаа нь тойргоор эргэх хөдөлгөөний хувьд ГАЛ алгоритм нь тооцоол илүү сайн хийдэг нь харагдаж байна. Зураг (7) дээр $\Delta = 0.1mm$ үед координат олох алгоритмуудын алдааг харьцуулан харуулсан байна. Энэ үед ГАЗ болон ГАЛ алгоритмын алдаа ойролцоо байгаа нь Δ хэмжээ бага авсанаас хамаарч байна. Хэдий муруй траектор зурагдсан хэдий ч маш бага хуваалтын хувьд шулуун гэж үзэж болно.

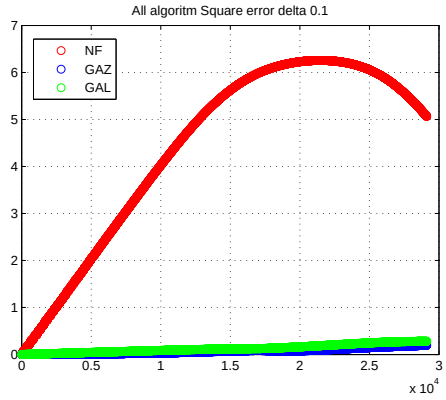
C. Олон муруй бүхий траектор

Олон муруйгаар илэрхийлэгдсэн муруйн хувьд даалгавар траектор үүсгэх нь алгоритмуудын онцлогыг тэдгээрийн алдаанаас харж болно. Муруй бүр нь олон төрлийн өнцөг, байгуулалт бүхий траектор болох юм. Бодит роботын хувьд ч гэсэн олон төрлийн муруй бүхий замын даалгавараар туулах хэрэгтэй болдог. Зураг (5) дээр дугуйн координатууд хэрхэн байгуулагдсаныг харж болно. Зураг (9) дээр $\Delta = 0.1mm$ үед координат олох алгоритмуудын алдааг харьцуулан харуулсан байна.

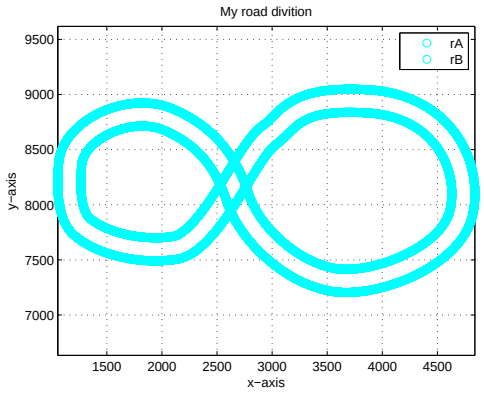
Олон муруй бүхий траекторын хувьд ГАЛ алгоритмын алдаа нь Δ -ын аль утганд бага байгааг нь харагдаж байна. Иймээс ГАЛ алгоритм нь залуурдагддаггүй тэнхлэг бүхий роботын координат олход илүү тохиромжтой.

V. Дүгнэлт

Алгоритмын харьцуулалт хийж туршилт хийхэд:
- Даалгавар траектор байгуулагч
- Траекторыг тэнцүү хэсгүүдэд хуваах алгоритм зэрэг олон ажилуудыг ашигласан. Зураг (3) дээр ГАЗ алгоритм нь ГАЛ алгоритмаас илүү нарийвчлалтай координат олж байгаа нь харагдаж байна. Бодит траектор байгуулалтын хувьд олон муруйгаас тогтсон траектор байдаг. Зураг (8) нь бодит траектор даалгаварын нэгэн жишээ болох юм. Зураг (8)-ын



Зураг 7: Алдааны харьцуулалт, $\Delta = 0.1mm$



Зураг 8: Олон муруй бүхий траектор даалгавар

хувьд алгоритмын харьцуулсан үр дүнгүүд нь зураг (9),(??), (11) дээрээс харж болно. Энэхүү харьцуулсан алдааны үзүүлэлтээс ГАЛ алгоритм нь алдааны хувьд багатайгаар координатыг олож байгааг харж болно.

Талархал

Энэхүү ажлыг одоогын үр дүнд хүрэхэд тусалсан багш М.Баярцүрэв /Доктор/, багш Д.Нанзадрагчаа /Магистр/, сургалтын инженер Гантулга нартаа талархал илэрхийлэё.

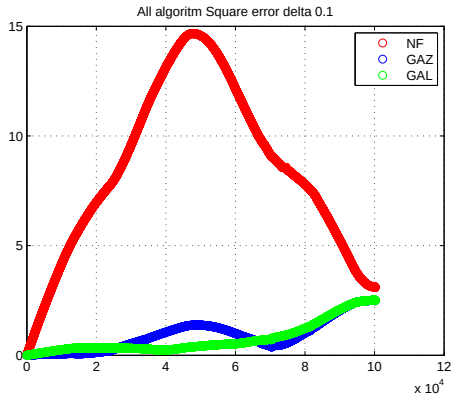
Ашигласан ном

[1] D.Lee, J.Villasenor, W.Luk, P.Leong ,"Mobile robot localization and object pose estimation using optical encoder, vision and laser sensors," IEEE Intern. Conf. on Automa. and Logist., pp. 617 – 622, 2008.

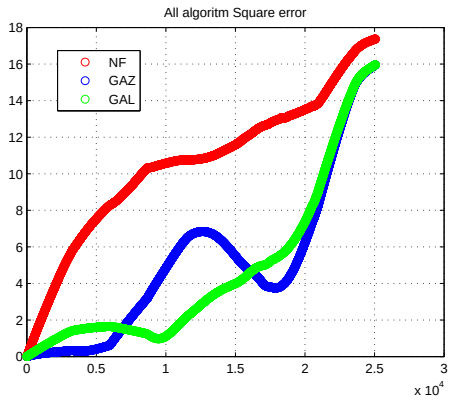
[2] Inigo Thomas, J.Oliensis ,"Automatic position estimation of a mobile robot,"Ninth Conf. on Artificial Intelligence for Applications., pp. 438 - 444 , 1993.

[3] Sungbok Kim , Sanghyup Lee,"Mobile robot localization and object pose estimation using optical encoder, vision and laser sensors," IEEE Conf. on Digital Object Identifier., pp. 1167 - 1172 , 2008.

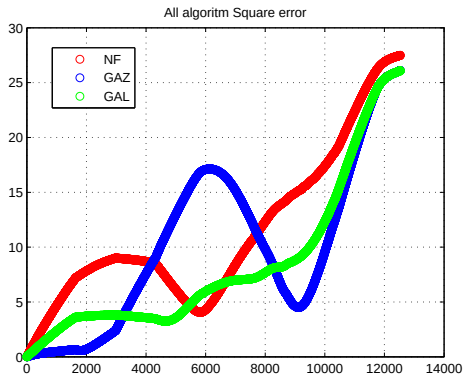
[4] Gao Qingji, Feng Qi, Zhang Hongxiang,"A robot self-localization method based on perception dominance," IEEE Intern. Conf. Digital Object Identifier., pp. 2343 – 2346, 2011.



Зураг 9: Алдааны харьцуулалт, $\Delta = 0.1mm$



Зураг 10: Алдааны харьцуулалт, $\Delta = 0.4mm$



Зураг 11: Алдааны харьцуулалт, $\Delta = 0.8mm$