# OTONOM EKSKAVATÖR SİSTEM ARAŞTIRMASI VE TASARIMI

## 1-Otonom araçlar hakkında literatür araştırması:

Otonom hareketli araçların genel yapısı, kontrol özellikleri, algılama sistemleri, lokalizasyon sistemleri ve hareket algoritmaları hakkında araştırmalar yapıldı. Bunların sonucunda araçta kullanılacak temel sensörler belirlendi.

* Lidar
* Kamera
* Encoder (Teker odometrisi)
* GPS
* İmu

Sensörlerden gelen verilerin okunması, işlenmesi ve kontrol sisteminin uygulanması için Ros tabanında çalışılmasına karar verildi.

Ros ile ilgili literatür araştırmaları sonucun da:

* Gps verisinin hassasiyetinin arttırılması için GNSS-Rtk gibi bir iyileştirmeden yararlanılmasına karar verildi.
* Gps, İmu ve Odometri verileri benzer özellik bilgileri (Hız, konum, ivmelenme gibi) sağladıkları için birlikte optimize kullanımları üzerine çalışma yapılacak.
* Ekskavatörün kendi etrafını algılayıp yakındaki engel ve tehlikeleri bildirmesi için Lidar’dan yararlanılacak.
* Ros da hareket algoritmaları üzerine çalışalacak, sensörlerin optimizasyonu tamamlanınca ekskavatörün önceden belirlenmiş bir alanda tanımlanmış hedefe otonom olarak hareketi simülasyonda test edilip en doğru hareket algoritması tercih edilecek.

## 1-2-GNSS+RTK

Gerçek Zamanlı Kinematik anlamına gelen Real Time Kinematic (RTK), bir baz istasyonunun yakınında yüksek konumlandırma performansı sağlayan farklı bir GNSS tekniğidir. Teknik, taşıyıcı ölçümlerin kullanımına ve konumu iyi bilinen baz istasyonundan geziciye aktarımlara dayanır, böylece bağımsız pozisyonlandırmayı yönlendiren ana hatalar iptal edilir. Bir RTK baz istasyonu yaklaşık 10 veya 20 kilometre yayılan bir servis alanını kapsar ve baz ile gezici bağlantı için gerçek zamanlı bir iletişim kanalı gerekir. Birkaç santimetre aralığında performans sağlamaktadır.

Baz istasyonu, bilinen tüm konumlarını, tüm uydular için L1 ve L2 frekanslarında kod ve taşıyıcı ölçümleriyle birlikte yayınlar. Bu bilgi ile gezici ekipman, faz belirsizliğini sabitleyebilir ve tabana göre konumunu yüksek hassasiyetle belirleyebilir. Temelin konumunu ekleyerek, gezici küresel bir koordinat çerçevesinde yerleştirilir. Yapacağımız çalışmada santimetre aralığında hassasiyet gerektiği için RTK tercih edilmiştir.

Ros tabanında GPS,İMU ve Odometri verilerinin kullanımı ile ilgili araştırmalar sonucun da, Roscon 2015’de açıklanan [1] Robot\_localization yapısının kullanılması gerekmektedir. Bu sistemlerin Ros ortamında test edilebilmesi için Gazebo ortamında bir robot modeli olusturuldu.

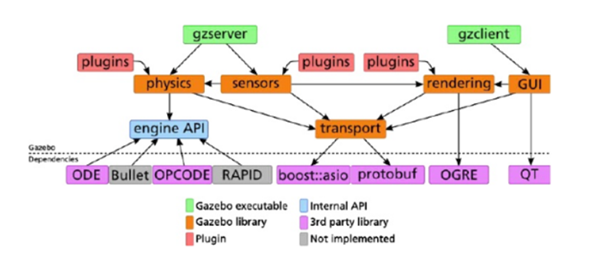
## 2-Ros

ROS, açılımı Robot Operating System olan ve robotları kontrol etmeyi sağlayan bir yazılımdır. İsminde işletim sistemi ifadesi geçse de insan ile robot arasında iletişimi sağlayan açık kaynak kodlu bir ara yüz yazılımı denebilir. Robot İşletim Sistemi (ROS), robot yazılımı yazmak için esnek bir çerçevedir. Çok çeşitli robotik platformlarda karmaşık ve sağlam robot davranışı yaratma görevini basitleştirmeyi amaçlayan araçlar, kütüphaneler ve sözleşmeler koleksiyonudur

## 2-1-Gazebo ve Rviz

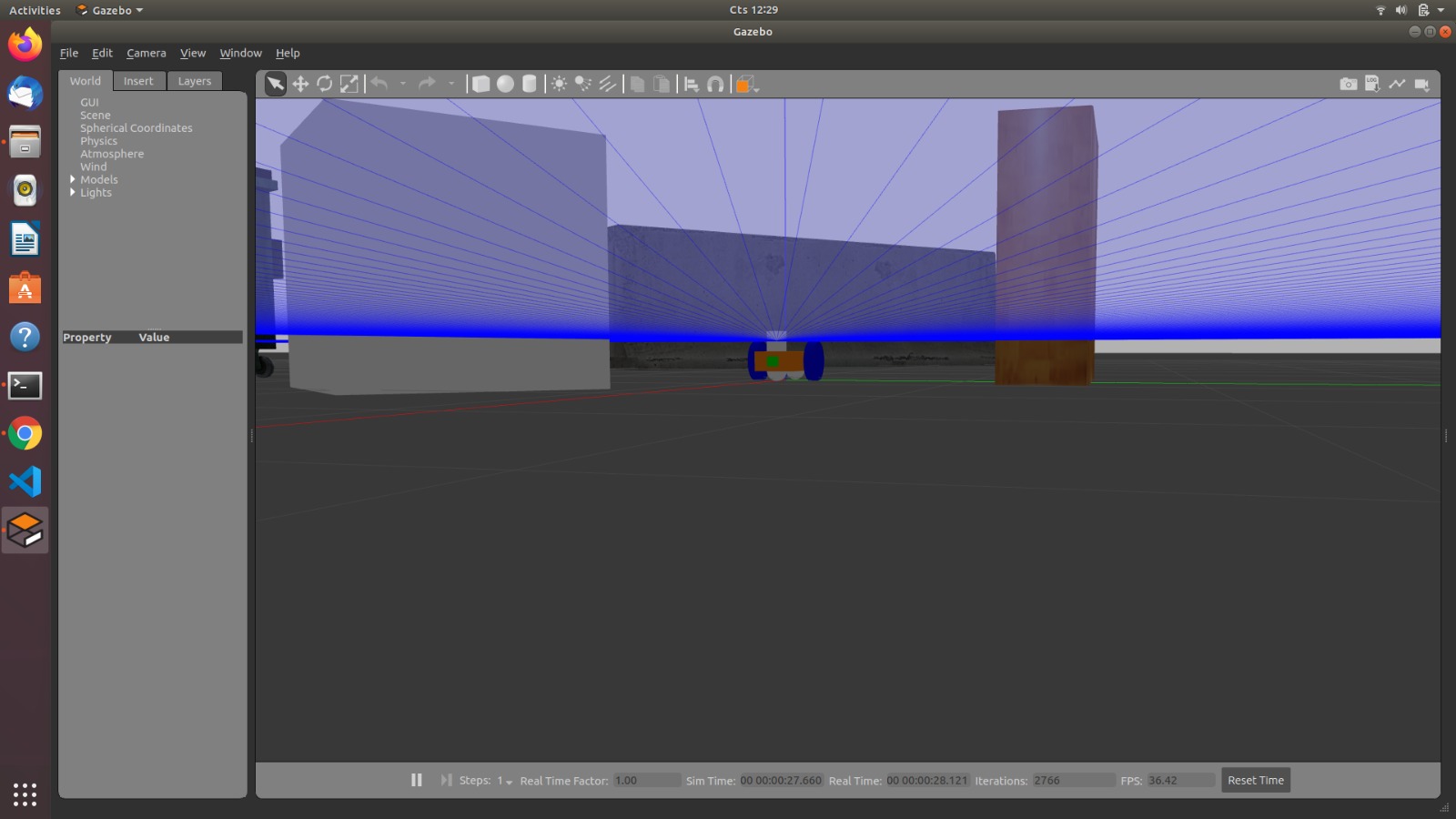
Gazebo ücretsiz ve açık kaynak kodlu, robot simülasyon ortamıdır. Bir ve birden fazla robotun 3 boyutlu ortamda simüle edilebildiği bir programdır.

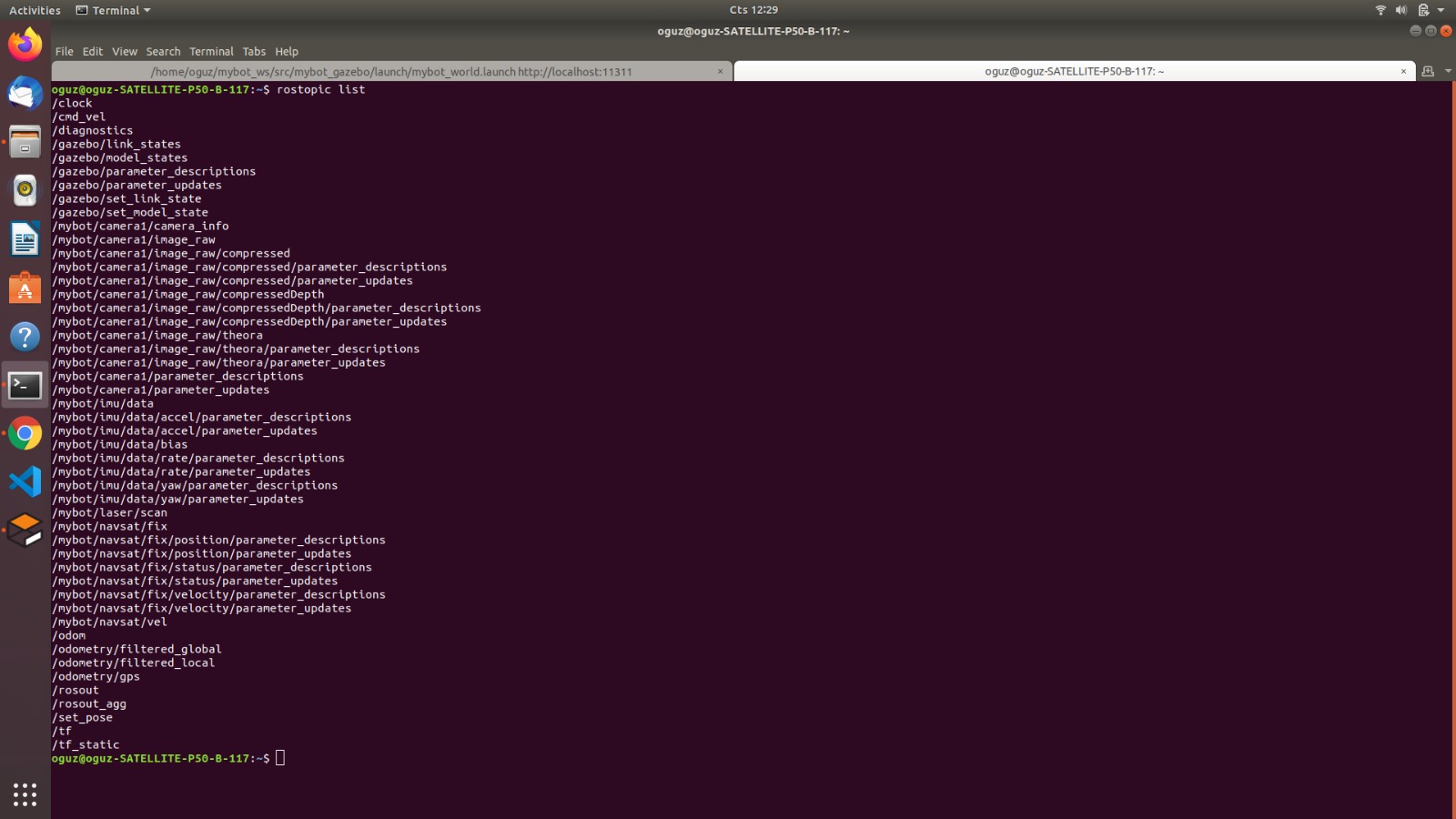
Gazebo genelinde server-client mantığıyla çalışmaktadır. Gzserver (sunucu) somut işlevleri yaparken (robot ortamda dolaşırken), Gzclient (istemci) kullanıcının isteklerinin yerine getirilmesi ve simülasyonun görsele aktarılmasını sağlar. Kendi robot ve dünyamızı yaratma imkânı bize tanınmıştır. Eğer istenilirse model.config ve model.sdf dosyalarının içindeki XML kodlarını değiştirerek fiziksel robotlara istenilen sensör ve kamera ekleyebilme şansımız vardır. Oluşturulan bu modeller için işletim sisteminin SDF ve URDF formatlarını desteklemesi gerekmektedir. Diğer simülasyon programlarına nazaran Gazebo, sensör ekleme ve gerçeklenmesi konusunda daha fazla verimlidir. Sersörlerden alınan verilere gürültülerin dâhil edilmesi gerekmektedir. Gerçek dünyada gürültüsüz veriye ulaşılması imkânsızdır. Bu sebepten dolayı simülasyon ortamının gerçek dünyadaki gibi gürültülü veri üretmesi sağlanarak ideal robot tasarımı gerçeklenmelidir. Gazebonun sensörleri sayesinde gürültülü veri oluşturmak mümkündür.



Rviz, ROS için bir 3D görselleştirme ortamıdır. Rviz, robotun ne gördüğünü, ne yaptığını görmemizi sağlar. Sensör bilgilerinin görselleştirilmesi ve kaydedilmesi, geliştirme ve hata ayıklamada önemli bir parçasıdır.

Gazebo da model oluşturulurken [2] kaynağından yararlanılmıştır. Modelimize Camera, Lidar, GPS, İMU, Wheel odometry blgileri eklenmiştir.

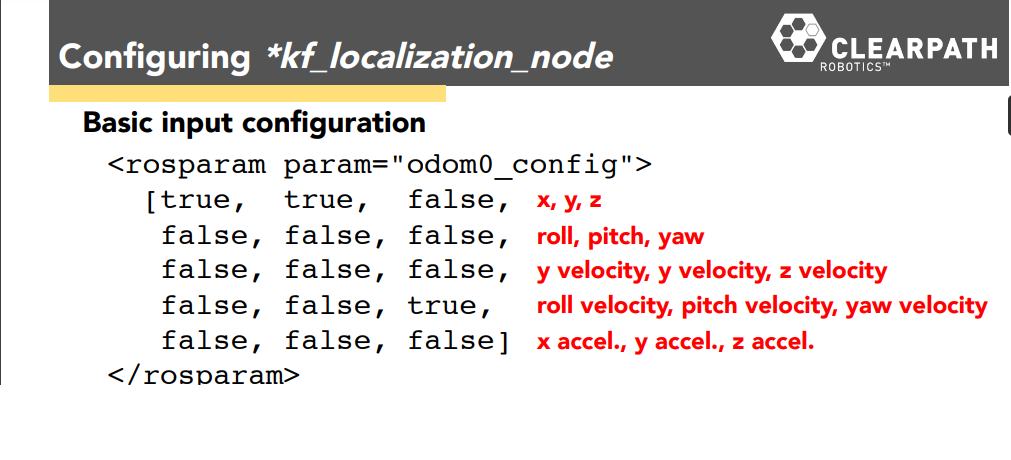
Resim-1 Gazebo ortamı

Resim-2 Rosmasterda çalışan Topicler

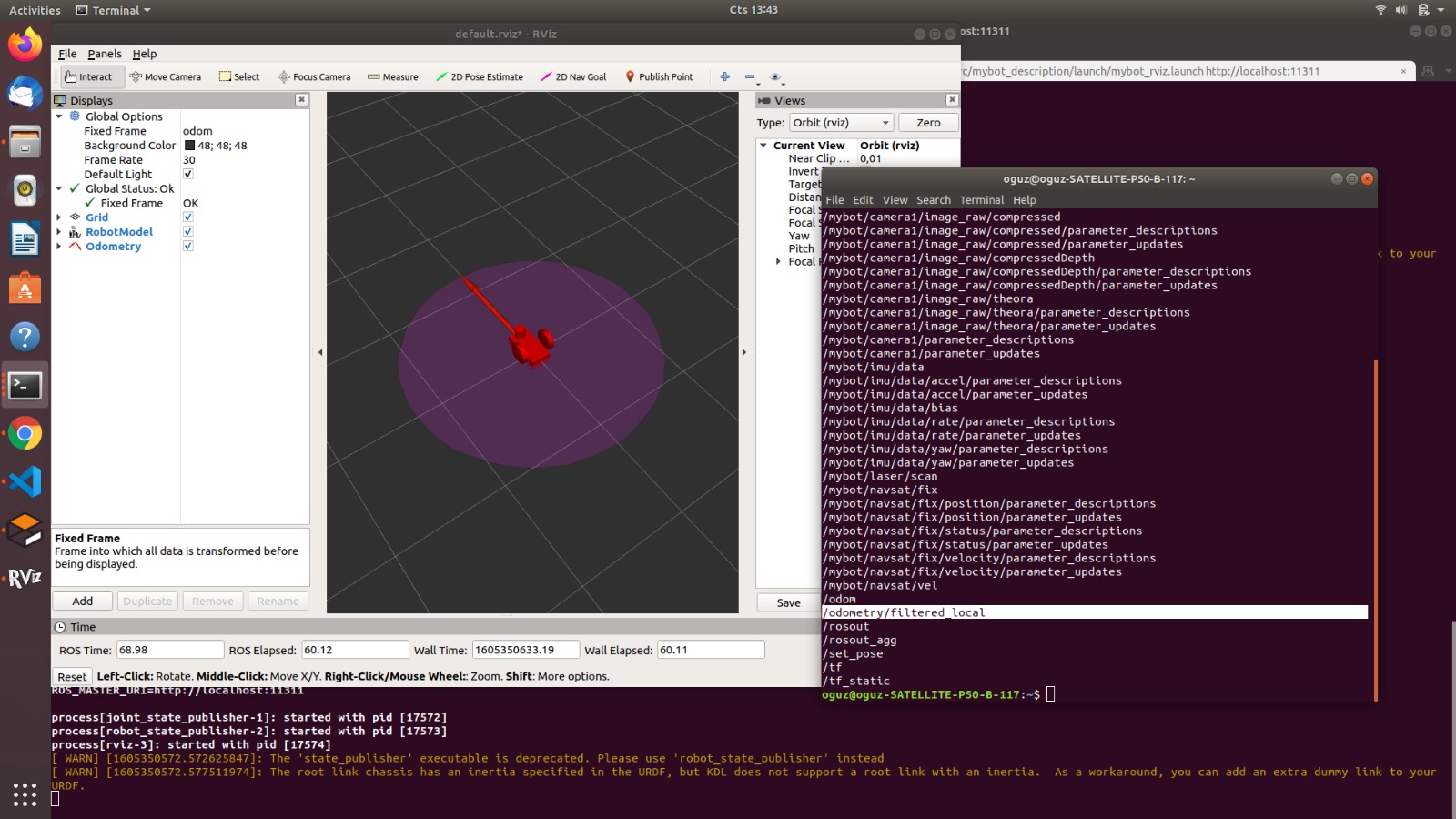
## 2-1-Robot\_Localization (Wheel odometry – İmu – Gps verilerinin birleştirilmesi)

Sistemde kullanılan 3 sensörde belli noktalarda aynı durum için ölçümler yapar. Bu sensörleri ayrı ayrı kullanıp işlemek hem hata payı anlamında hemde işlevsellik anlamında bize büyük engeller oluşturmaktadır. Bu nedenle Robot\_localization paketi ile bu sensörlerin ölçümlerinin bir kalman filtresi yardımıyla birleştirilip daha doğru sonuçlar elde etmeyi amaçlamaktayız. Gps tarafında baktığımızda da gelen latitude ve longitude . verilerini kendi sahip olduğumuz frame yapısına doğru bir şekilde dönüştermek için navsat\_transform yapısını kullanacağız.

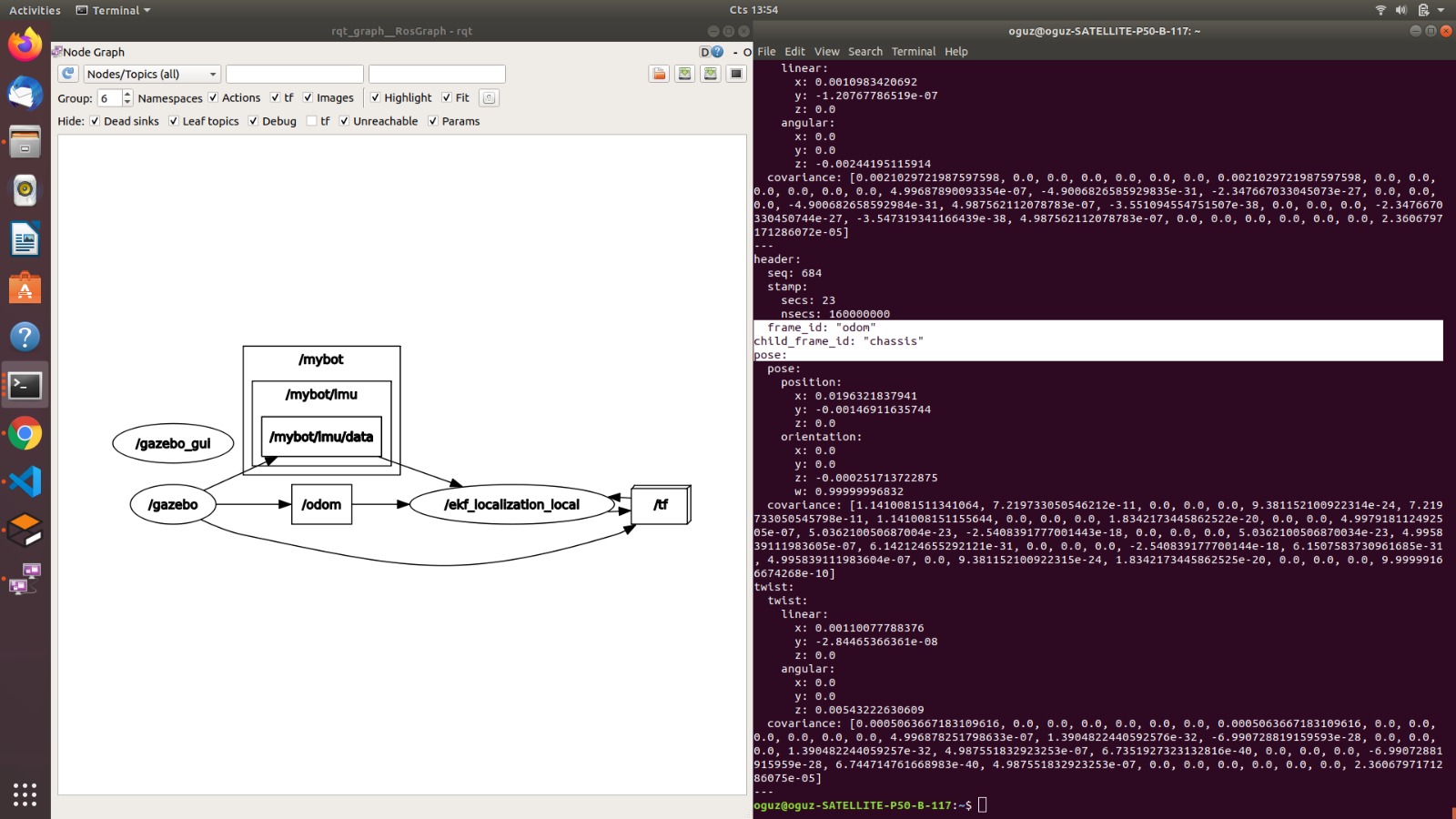
Robot\_localization paketi temelde EKF\_node (extenden kalman filter) yapısını kullanır. Ana mantığından bahsedecek olursak, aracımızla ilgili 15 parametreli (resim 3) bir çıkışa sahip olan bi topic yayınlar. Giriş olarak da bir den fazla sensör verisi kullanır (Bizim çalışmamız için bunlar /odom - /imu/data - /odometry/gps ), biz bu sensörlerden istediğimiz ölçümü seçerek toplamda (genelde bu topic üzerinden odometry/filtered ) 15 ölçüm değerine sahip bir çıkış alırız.

Resim-3 Ayarlanan 15 değer

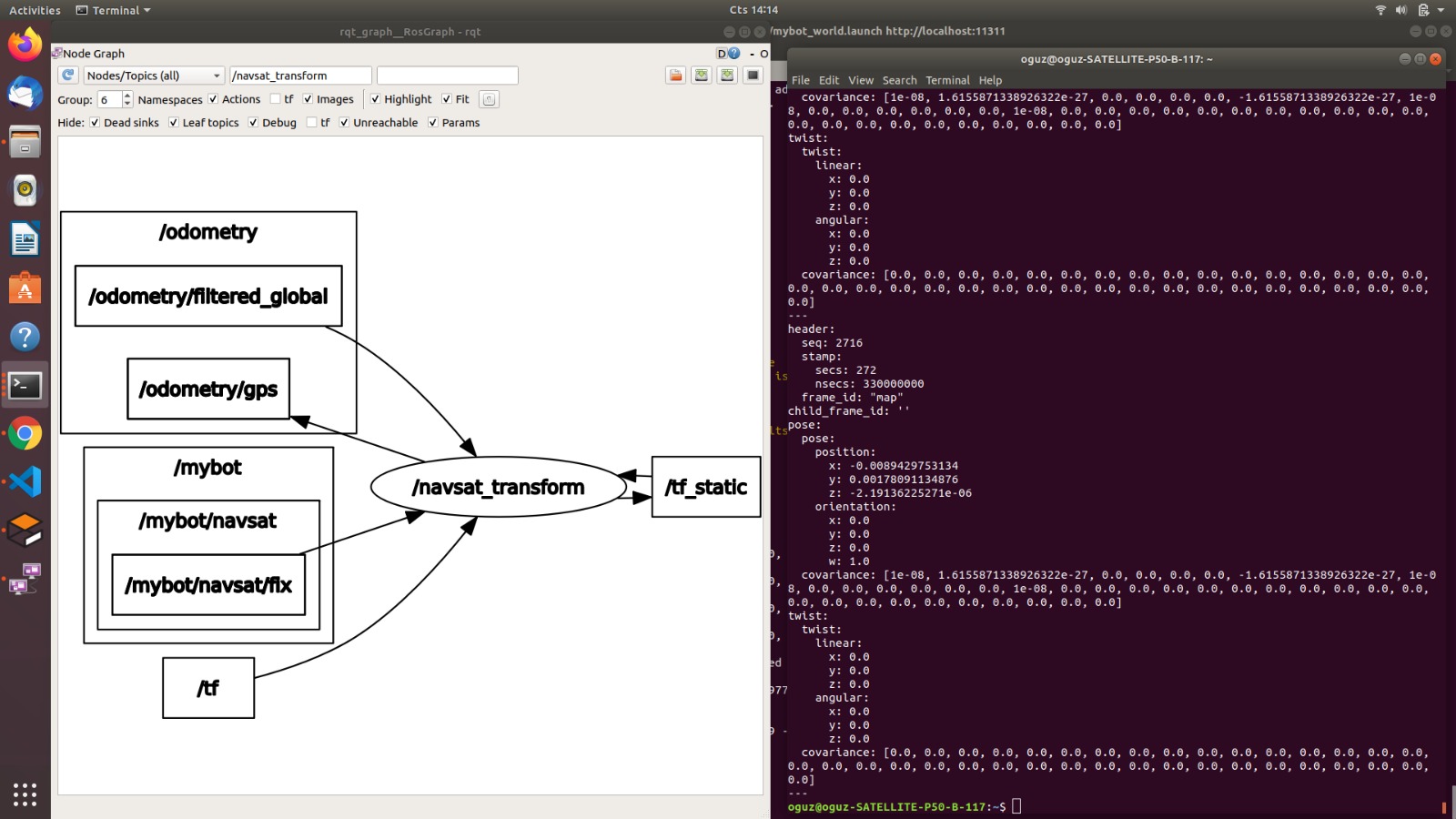
[1] ve [3] baz alınarak kendi modelimize göre bu parametreleri ayarladık. Başlangıc da yapının anlasılabilmesi için sadece odometry ve imu verisiyle başladık. (Resim-4)

Resim-4 Ekf\_local’den üretilen odometry/filtered\_local gözükmektedir.

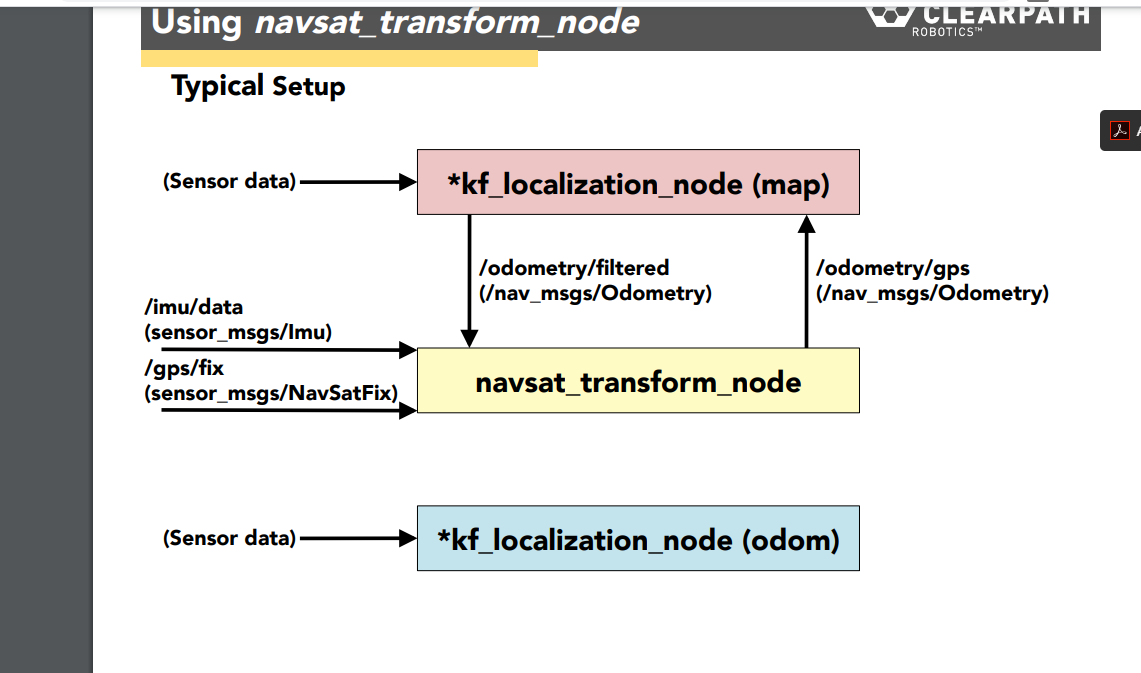
Bu aşamada odom --> chassis dönüşümü yapılmış ve gerekli çıkış elde edilmiştir. Burada odometry/filtered\_local dinlendiğinde akan veriler (Resim-5) ve alt tarafda bağlantılı Topicler bulunmakta.

Resim-5 ekf\_local

Bir sonraki aşama olarak gps verisinin bu filtreye eklenmesi lazım. Ancak biz gps verisi olarak enlem ve boylam verisi alıyoruz bu veriyi kendi dünya çevremize yani şuanki hali ile odom çerçevesinin konum bilgisine dönüştürmeliyiz. Bunun için [4] navsat\_transform\_node yapısını kullanacağız. Bu yapıda giriş olarak imu/data – ekf’den üretilen odom – navsat/fix (enlem boylam bilgisini içeren gps) çıkış olarakda odometry/gps (dönüşümü yapılmıs gps bilgisi) elde edeceğiz. [4] kaynağını incelerseniz imu verisinin bu dönüşümün doğru yapılması için önemlidir.

Resim-6 Navsat transform

Resim-6’da ekranın sağ tarafında odometry/gps dinlendiğinde dönüşümün yapıldıgı görülmektedir. Bu aşamada katman filtremize odometry/gps verimizide ekleyeceğiz ancak [5] incelendiğinde 2 adet katman filtresi kullanmamız gerektiği görülmektedir. Veri bir taraftan yenilenen bir akıma sahip olduğundan tek katman filtresinde yapısal olarak atlamalar olacaktır. 2 ekf üzerinden çalışarak bir tarafı sürekli bırakıp yenilenen veriyi 2. dosyada gerceklestirdik. Burdada ilk dosyada odom --> chassis dönüşümü yapılırken 2. dosyada map --> odom dönüşümü yapacağız.

Resim-7 Dosyalar arası etkileşim

[1] https://roscon.ros.org/2015/presentations/robot\_localization.pdf

[2] http://gazebosim.org/tutorials/?tut=ros\_urdf

[3] http://docs.ros.org/en/melodic/api/robot\_localization/html/\_downloads/robot\_localization\_ias13\_revised.pdf

[4] http://docs.ros.org/en/melodic/api/robot\_localization/html/navsat\_transform\_node.html

[5] http://docs.ros.org/en/melodic/api/robot\_localization/html/integrating\_gps.html