Borsa di studio attivata ai sensi di quanto disposto dal D.M. n. 1061 del 10/08/2021

Titolo del progetto: Logistica efficiente mediante robot mobili energy-aware

La borsa sarà attivata sul seguente corso di dottorato accreditato per il XXXVII ciclo:
INGEGNERIA INFORMATICA

Responsabile scientifico: Giorgio Grisetti
Area per la quale si presenta la richiesta: GREEN
Numero di mensilità da svolgere in azienda: 6
Numero di mensilità da svolgere all'estero: 6 presso Universita' di Bonn
Azienda: Bosch Research center (Renningen/DE)
Il Dipartimento è disponibile a cofinanziare per un importo pari a euro: 10000
Dipartimento finanziatore: DIPARTIMENTO DI INGEGNERIA INFORMATICA, AUTOMATICA E GESTIONALE - ANTONIO RUBERTI- con delibera del 20/9/2021

Progetto di ricerca:
La capacita' di muoversi in modo autonomo in un ambiente e' riconosciuta come un aspetto fondante per numerose applicazioni che coinvolgono sistemi robotici. Sistemi quali aspirapolveri e tagliaerbe automatici, sistemi per la movimentazione l'automazione di magazzini, manipolatori mobili e non ultimo automobili a guida autonoma sono oggi realta' e si prevede che il loro impiego crescera' nel futuro.

La moderna societa' fortemente digitalizzata e' divenuta sempre piu' affamata di energia e le persone sono spesso non consapevoli del reale cosumo del software che usano. Un singolo tab del browser aperto su una pagina di una certa complessita' puo' comportare il consumo di dieci o piu' watt, nel caso in cui la piattaforma usata sia un PC desktop. In pratica un tab del browser e' energeticamente equivalente all'accensione di una lampadina ad alta efficienza energetica. Ciononostante pochi utenti si preoccupano del numero di tab aperti, o di usare il computer in modo efficiente.

Il consumo energetico del software tende a crescere con il tempo, nonostante gli sforzi dei produttori di hardware nell'aumentare l'efficienza dei dispositivi. Gli sviluppatori software si aspettano di veder crescere la potenza computazionale a loro disposizione, in maniera illimitata. Poiche' il mercato del software e' dinamico ed i prodotti sono soggetti a continue e costanti modifiche ed aggiustamenti per integrare nuove funzionalita' su richiesta del cliente, una tendenza comune e' quella di sviluppare sistemi che internamente eseguano molto piu' di quanto richiesto. Nella mia esperienza di professore universitario ho assistito a studenti che utilizzano strumenti spesso esagerati nella soluzione di un problema.

Esempi includono l'inclusione di un database per gestire dieci utenti con nome e cognome, o l'uso di librerie di algebra lineare su GPU per eseguire una singola moltiplicazione di matrici tre per tre.

In sostanza programmi inefficienti - laddove non siano fatte scelte algortmiche sbagliate - sono spesso il risultato di una scelta alla radice che favorisce bassi tempi di sviluppo alla qualita' e velocita' del prodotto finale, nonche' nell'abuso di tecnologie non necessarie.

Sotto questo aspetto, i sistemi di navigazione autonoma non sono diversi dal resto del software. Inoltre va considerato che tali sistemi sono in controllo di attuatori e sensori che movimentano una piattaforma robotica e che contribuiscono in modo significativo al consumo del sistema. Inoltre, il modo in cui tali dispositivi sono controllati puo' impattare in modo significativo sull'energia spesa nonche' sull'usura dei dispositivi. Come esempio di ""abuso di computazione"", si riporta lo stack di navigazione di default del sistema ROS (Robot Operting System), che consiste di un localizzatore ed un pianificatore di traiettorie locale e globale. Tale sistema richiede sostanziali risorse computazionali per produrre sostanzialmente lo stesso risultato di software analoghi (e.g. Carnegie Mellon Navigation Toolkit - CARMEN) che nei primi anni duemila giravano su computer ben meno potenti di quelli odierni.

Parlando di sistemi commerciali, la situazione puo' essere non migliore. Al momento solo societa' abbastanza grandi sembrano avere le risorse per proporre soluzioni di navigazione autonoma su larga scala. In tali realta' la produttivita' di una squadra di programmatori e' spesso misurata piu' in termini del numero di linee di codice prodotte che non la densita'/qualita' delle stesse e dell'efficacia del risutato finale. Un altro esempio e' fornito da una ben nota camera di profondita', che viene fornita con un driver open source supportato dal produttore. Tale driver consta in diverse migliaia di righe di codice, e richiede risorse rilevanti. Nel 2010 uno studente implemento' un modulo per il kernel di Linux in poche migliaia di linee che permetteva l'integrazione ""trasparente"" del dispositivo nel sistema operativo, riducendo cosi' il numero di componenti accessorie. Inutile dire che tale driver risultava notevolmente piu' efficiente dell'originale permettendo con poche modifiche l'uso di tale camera su processori a ARM consumo relativamente basso, in contrasto con CPU Intel di fascia medio-alta richiesti dal driver originale.

Come detto in precedenza, una delle ragioni principali per l'inefficienza energetica del software e' conseguenza di un approcchi allo sviluppo di programmi che rammenta il comporre componenti ciascuno dei quali e' in gradi di affrontare un problema di gran lunga piu' complesso rispetto a quello che deve effettivamente risolvere.

Uno degli aspetti che appare in ogni sistema di navigazione e' la mappa. E' chiaro che la mappa e' necessaria per la navigazione autonoma, poiche' senza mappa lo stesso concetto di posizione del robot viene meno del suo significato. Analogamente, e' chiaro che una mappa deve possedere sufficiente informazione affiche' un robot possa ""riconoscere"" gli ambienti di tale mappa attraverso i sensori di cui e' dotato per determinare la sua posizione nell'ambiente. Inoltre al fine della pianificazione delle traiettorie, la mappa deve contenere informazioni di supporto quali regioni traversabili o aree proibite.

Quasi tutti i sistemi di robot mobili autonomi proposti finora all'utente finale comprendono la funzionalita' per ""apprendere"" la mappa dai dati sensoriali, attraverso qualche forma di sistema di SLAM (Simultaneous Localization and Mapping). E' da notare che SLAM e' notevolmente piu' complesso - e computazionalmente oneroso - rispetto alla pura localizzazione. Cio' deriva dal fatto che poiche' la dimensione delle quantita' da stimare nel primo caso (mappa e posa del robot) e' notevolmente maggiore che nel secondo caso (solo posa del robot). In ogni caso, una volta che una mappa e' stimata un robot mobile puo' effettuare missioni nell'ambiente solamente avvalendosi di localizzazione e planning, fino a quando la mappa rappresenta in modo sufficientemente fedele la realta' di interesse.

Molti sistemi commerciali affrontano il problema dell'obsolescenza della mappa eseguendo costantemente un processo di stima della mappa, in modo da operare su una stima aggiornata della stessa. Questa scelta ha effetti sull'hardware devoluto alla computazione, poiche' deve essere in grado di eseguire algoritmi piu' complessi rispetto al caso in cui debba solo eseguire localizzazione, con conseguenze sul consumo energetico. Inoltre, nel caso di una installazione multi robot - quale potenzialmente quella di un sistema di logistica automatizzato - l'intero sistema perderebbe il concetto di stato/mappa globale, poiche' ogni piattaforma ha una stima sostanzialmente indipendente dalle altre. Avere un sistema in grado di centralizzare e fondere tutte le misure della flotta di robot al permetterebbe di ottenere una singola globale e consistente stima dello stato. Su questa stima algoritmi di pianificazione potrebbero considerare aspetti come la congestione in alcune parti della mappa, sue variazioni, presenza di ostacoli al fine di prendere le misure necessarie per ottimizzare una funzione di costo globale. Quest'ultima, in questo caso considera non solo il tempo di esecuzione ma anche l'energia consumata. Il sistema potrebbe decidere di ritardare la partenza di un robot, in attesa che una congestione venga risolta, assegnargli un altro task oppure intraprendere una missione su una via alternativa.

Come ulteriore sviluppo, lo sviluppo di un planner di alto livello potrebbe essere incaricato di assegnare le missioni ai componenti

della flotta, in base alla situazione dell'ambiente attuale e predetta. Infine, per un sistema multi robot che operi su larga scala ed in ambienti misti indoor/outdoor, si potrebbe considerare un ulteriore livello di mobilita' che ""emuli"" le reti di trasporto pubblico delle nostre citta'. In tali contesti si possono immaginare veloci ed efficienti robot per esterno che possano trasportare piu piccoli ed agili robot per interno. Poiche' sia le posizioni che le intenzioni dei robot sono a disposizione del sistema di pianificazione, le corse dei robot outdoor possono essere modulate dinamicamente per ottimizzare l'efficienza.

Per concludere, in questo progetto proponiamo lo studio e lo sviluppo di sistemi multi robot per la stima consistente e centralizzata dello stato dell'ambiente, e sia in grado di operare in tempo reale. Inoltre analizzeremo lo sviluppo di sistemi di navigazione efficienti sia dal punto di vista energetico e computazionale, avvalendoci di uno sviluppo hardware-software integrato. A tale scopo ci avvarremo dell'esperienza acuisita nello sviluppo del robot mobile MARRTINO, correntemente un uso in diverse scuole dep Paese, come ausilio all'insegnamento di materie STEM. Valideremo il nostro sviluppo mediante l'uso di simulatori in una prima fase, per poi passare ad esperimenti con piattaforme reali. Dissemineremo i risultati della ricerca e dello sviluppo mediante pubblicazioni su conferenze e riviste internazionali, e distribuiremo il software ed i progetti dell'hardware attraverso licenze open source. In nostro fine ultimo e' fornire un infrastruttura software ed una piattaforma di riferimento hardware che possano supportare gli attori sul mercato nell'uso efficiente ed efficace di sistemi di navigazione autonoma.

Le metriche di successo delle nostre ricerche saranno misurare in termini di energia risparmata rispetto a sistemi correnti allo stato dell'arte nel nel compiere le stesse missioni. Sebbene un risparmio di poche centinaia di watt per piattaforma possa sembrare non rilevante, e' nostra convinzione che il numero di robot mobili crescera' nel futuro, e con essi il risparmio energetico indotto dal lavoro presentato in questa proposta. Per concludere crediamo che alcuni aspetti della nostra ricerca possano essere un utile complemento a studi sulla mobilita' sostenibile. Il sistema da noi sviluppato, operando in condizioni di completa conoscenza dello stato e delle intenzioni degli agenti ma al tempo stesso essendo soggetto ad incertezze che derivano dall'operare nel modo reale puo' rappresentare un valido strumento di sperimentazione, analisi e simulazione in tali dominii.

Titolo del progetto (inglese): Towards effective logistics by energy aware autonomous navigation

Progetto di ricerca (inglese):

The ability to autonomously move in the environment is regarded as an essential building block for several robotic applications that are supporting our everyday life and the production processes. Autonomous vacuum cleaners and lawn mowers, drones for pick and delivery and surveillance, mobile manipulators in industry and logistics and not ultimately self driving cars are only a few systems that rely on autonomous navigation.

In parallel, the current internet oriented society have become power hungry, and users are often unaware of the true power consumption of the software they are using. Opening a browser tab on a complex webpage might absorb 10 or more watts, if the client is a desktop PC. To make a parallel, opening a browser tab is equivalent to turning on an energy efficient light bulb in a room. Yet few of us care of the number of tabs or other aspects.

This occurs despite the efforts put by the hardware manufacturers in decreasing the power efficiency of their devices. Software developers have been used to see the available computational power constantly increasing. Since the software market is highly dynamic where products are constantly patched and modified to fulfill the changing customer's needs in short time, a common trend is to provide systems that ""under the hood"" does much more than what is needed. In my life as a professor I have seen students using large tools where a naive direct implementation would suffice. Examples include a database engine in their design to handle a list of 10 users, or linking a large GPU math library to do a single multiplication of three by three matrices.

Overall, inefficient programs are mainly due to the choice of favoring short development times over quality and speed of the resulting software through the inclusion of non necessary components.

Autnomous navigation systems are no different from other software in this respect. Additionally, navigation systems control actuators to move a robot platform, and these might drag far more power than the software itself. The way these motors are controlled has a high impact on the overall consumption of the system and on its mechanical wear. As an example of computation misuse consider that to effectively run the ""standard"" navigation stack provided by ROS (Robot Operating System), (localization, global and local path panning), one needs significant computational resources. The same task, with even less accurate sensor data was carried on in the early 2000 with far less resources.

When it comes to commercial systems, the situation might not be better. Only big companies have the means to address these problems on a large scale. In these realities, the productivity of a team is often measured in the number of lines of code produced, and not

necessarily on the density/quality and effectiveness of the resulting solution. As an example, a popular depth camera comes with an open-source driver developed by the manufacturer and spans over several thousands lines of code. In the late 2010 a student implemented a kernel-level driver that seamlessly integrates this camera on a Linux system, and it was no more than a couple thousand lines of code. More efficient and more compact, with minor modifications this alternative driver allowed to control one of these cameras from an ARM platform while the ""official"" release system required a mid-end Intel PC.

As stated before in this document, one of the main reasons of software inefficency is the trend of designing systems by putting together components that approach problem far larger than those for which they are required. One of the components that appears in every navigation system is the map. It is obvious that a map is required for navigation, since without it the whole concept of ""robot position"" has no meaning. Similarly, it is clear that a map should possess enough information to be ""matched"" by the robot on-board sensors, otherwise it would not be able to localizing in it. Furthermore, to the extent of planning, the map should contain information to carry on the task, such as ""traversable areas"" and ""no go zones"". Almost all navigating robots that have been proposed so far therefore implement the functionality of ""learning"" the map from the sensory data, through some sort of Simultaneous Localization and Mapping (SLAM) algorithm. We remark that SLAM is a way more complex problem than localization, since the dimension of the quantities to be estimated (the map and the robot pose) is far larger than the pose alone to be computed by localization. Yet, once the map is done, the system can go ahead with pure localization, and can reliably perform its missions as long as the environment is sufficiently represented by the map.

Most commercial systems, however tackle this aspect by continuously running a map estimation process, to operate on the most up-to-date map. This has some effects on the computational hardware that these systems should carry, which has to be more powerful than if localization only was performed, and drains more power. In addition, in case of a multi-robot installation the whole system looses the concept of global state/map, since each platform has its own. Having a centralized system that integrates the sensor measurements (or some data derived from them through on-board computation), would allow for a global and coherent update of the environment status. Planning algorithms might consider information such as the congestion in a certain part of the building in their calculation, similar to what modern car navigation systems do, and take the necessary measures to optimize the cost function. The latter, in this case should not only contain the execution time, but also the consumed energy. The system might decide if it is convenient to delay a mission or to start it

by taking a longer path.

Going one step further, a coarse notion of global planner might be in charge of assigning the missions to the multi-robot system based on the actual and on the predicted environment situation. Finally, for large multi robot systems operating in mixed indoor/outdoor scenarios, one might consider adding another layer that mimics the public transport system of our cities. In this context one might think about fast, energy efficient outdoor robots that act as ""buses"" for small indoor robots, that move from building to building using these ""vectors"". Since the position and the intentions of the small robots are known (and can be assigned), the schedule of the vectors can be dynamically modified to minimize the global cost function.

In summary, in this project we propose to investigate a global map maintenance system, that runs in a centralized or semi-centralized fashion and allows for real time updates. Furthermore we will investigate the design of resource efficient navigation systems, through a vertical integration and hardware-software codesign. To this extent, we already deployed some popular mobile platforms that are currently used by schools to support the teaching of STEM subjects. Furthermore we will investigate first bAttendo suey using simulators, then by using real platforms potential global multi robot planning systems, targeted at reducing the cost. We will disseminate the results of the research both in international conferences and journals and by open sourcing the developed software. We aim at providing an infrastructure and tools that support companies who want to leverage on this technology.

The success metrics of our findings will be measured in terms of power saved to accomplish the same task with traditional systems. Albeit saving ten to a few hundred watts per platform might seem little considering the small number of mobile robots that are currently on the scene, we expect this number to increase significantly in the near future, together with the energy saving. Furthermore, we believe that our research might pave the way for future plans of sustainable mobility by providing a mocked application where the intentions and the location of all users are known.