2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Autonomous Door Opening and Plugging In with a Personal Robot

Wim Meeussen, Melonee Wise, Stuart Glaser, Sachin Chitta, Conor McGann, Patrick Mihelich,
Eitan Marder-Eppstein, Marius Muja, Victor Eruhimov, Tully Foote, John Hsu, Radu Bogdan Rusu, Bhaskara Marthi,
Gary Bradski, Kurt Konolige, Brian Gerkey, Eric Berger

Willow Garage, USA
{meeussen,mwise}@willowgarage.com

Abstract— **We describe an autonomous robotic system capable of navigating through an office environment, opening doors along the way, and plugging itself into electrical outlets to recharge as needed. We demonstrate through extensive experimentation that our robot executes these tasks reliably, without requiring any modification to the environment. We present robust detection algorithms for doors, door handles, and electrical plugs and sockets, combining vision and laser sensors. We show how to overcome the unavoidable shortcoming of perception by integrating compliant control into manipulation motions. We present a visual-differencing approach to high-precision plug-insertion that avoids the need for high-precision hand-eye calibration.**

I. Introduction

We aim to develop robots that combine mobility and manipulation to assist and work with people in offices and similar institutional settings. These robots inhabit an environment that is simultaneously structured and varied; designed according to social and legal norms, but under constant change by the people who use it. To be effective in even a single application, such a robot must exhibit an array of capabilities, some specific to the application, but many with broader utility.

We focus on two basic capabilities that are necessary for most, if not all, applications: navigation and energy management. We want our robots to safely go anywhere that people can go, avoiding all obstacles, opening doors, and passing through doorways along the way. When low on energy, a robot should find a standard electrical outlet and plug itself in to recharge. A robot that can accomplish these tasks reliably, without requiring environmental modifications, represents a step toward long-running autonomous robots that will serve as a foundation for application development of all kinds.

In this paper, we present an implemented system comprising autonomous navigation, door-opening, and plugging-in. We have validated the robustness and effectiveness of this system through extensive experimentation, including a "challenge" in which the robot was required to plug itself into ten outlets in ten rooms in an office building, opening doors along the way. The contribution of our work is four-fold:

1) demonstration of an integrated mobile manipulation system that can robustly find doorways, enter rooms, and plug itself into standard sockets in unaltered indoor environments;
2) perception algorithms combining vision and laser sensors for robust detection of doors, door handles, and electrical plugs and sockets;
3) integration of compliant control into motion strategies to simplify manipulation problems and overcome shortcomings of perception; and
4) a visual differencing approach to high-precision plug insertion, without a need for high-precision hand-eye calibration.

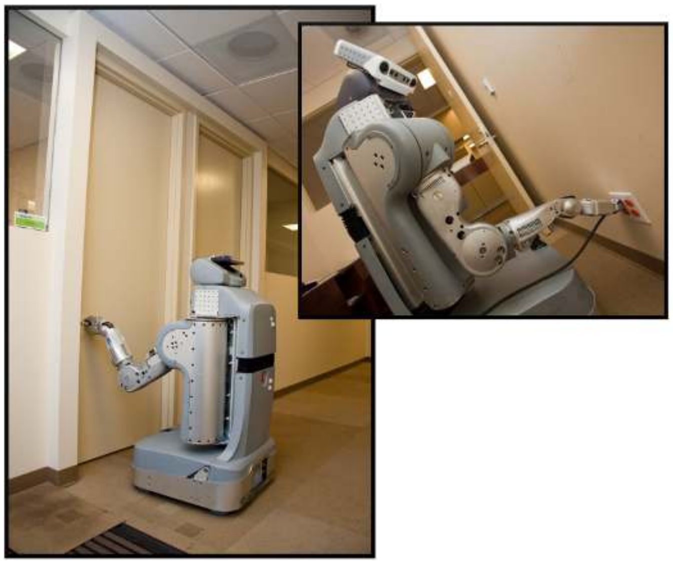


Fig. 1. The PR2 autonomously plugging in and opening a door.

The software described in this paper is available under an Open Source license,[1], and we encourage others to experiment with and use our code.

II. Related Work

A. Autonomous Recharging

Previously, researchers have taken two main approaches to the problem of long term task continuation and recharging with mobile manipulators:

[1]http://pr.willowgarage.com/wiki/doors_and_plugs

- modifying the working environment with a docking station [1], [2]
- detecting and plugging into a standard wall outlet in an unobstructed environment [3], [4], [5].

Docking stations are now commonly used as standard technology in household electronics [6], [7]; however, these robots are limited to unobstructed areas of the environment. If the charging station or outlet is in a room obstructed by a door, the robot will be unable to recharge and continue working. Many of these robots can recharge, but are designed for one specific task without generalization towards other tasks. In contrast, our approach uses a generalized mobile manipulation platform for navigation, door opening, and recharging, without specific design consideration for any single task.

B. Autonomous Door Opening

The door opening task can be broken into two parts: the detection of doors and door handles, and the opening of doors. In recent years, door and handle identification has been widely studied [8]–[12]. Each of these systems uses a single approach (either image, tactile or 3D data-based) to detecting doors and handles. In contrast, our system uses a combination of a laser perception-based approach (presented in detail in [13]) and an image-based approach (described later in this paper) to robustly estimate the location of the handle.

Attempts at solving the door opening problem date back more than a decade [9], [12], [14], [15]. The system developed in [11] uses a Cartesian impedance controller with a mobile manipulator to push doors open, and is probably the closest to our approach in the use of compliant controllers. It does not, however, make any attempt at collision avoidance for the base of the robot, or address doors in different states. In [16], online estimation of the door model was combined with a hybrid system model to open doors. In [17], compliant control was used to open doors using a mobile manipulator. In contrast to our approach, none of the aforementioned made an attempt to develop a robust, reliable system that can smoothly open doors and recover from failures.

III. Problem statement

We study the problem of enabling a robot to autonomously perform the following tasks in an unmodified indoor setting:[2]

- navigate to any reachable location;
- open doors and pass through doorways; and
- plug itself into standard electrical outlets.

In this paper we concentrate on the door opening and plugging in tasks. We require that the robot be capable of opening doors in any state, including latched, ajar, partially open, and completely open. We also require that the robot recognize when a door is locked and move on to its next goal. For plugging in, the robot should recognize when it has succeeded, and give up after a reasonable period of time if it cannot.

An instance of the problem is given by a list of outlets to visit. The robot must visit and plug into each of these outlets, in any order, opening doors as necessary along the way. Each outlet is identified by a key that can be used to retrieve the outlet's approximate pose from a map.

Experiments in this paper were carried out on an Alpha prototype of the `PR2` mobile manipulation platform (Figure 1). The `PR2` uses an 8-wheeled omni-directional base to navigate in wheelchair-accessible environments. Equipped with two 7 degrees of freedom compliant arms, the `PR2` is designed for compliant interaction with the environment.

For navigation, a Hokuyo `UTM-30` laser scanner is mounted at ankle height. A second Hokuyo laser scanner is mounted on a tilting platform at shoulder-height, providing a full 3D view. The `PR2`'s head is a pan-tilt platform equipped with a high resolution 5 megapixel (MP) camera and, two stereo camera pairs with different fields of view and focal lengths.

The software controlling the `PR2` comprises many different interacting components: hardware drivers, controllers, perception algorithms, motion planning, high-level planning, etc.

To facilitate the computational needs of these components, and to ensure the scalability of the system, a distributed computing environment is needed. To accommodate these communication and distribution needs, we use the open source Robot Operating System `ROS`[3] for distributed computation on `PR2`'s four dual-core computers. `ROS` components (called *nodes*) hide the complexities of transferring data between processes, regardless of whether the processes run on the same machine or not [18]. Similarly, to expedite computer vision tasks, we make extensive use of the Open Source Computer Vision library `OpenCV` [4] [19].

A. Supporting infrastructure

Beyond the basic hardware and software described in the previous section, we rely on two key pieces of supporting infrastructure: navigation and executive control.

a) Navigation: The `PR2` autonomously navigates in an indoor office environment using a 2D occupancy grid map built from base laser scans, with 2.5 cm grid resolution [20]. This system combines probabilistic localization, 3D obstacle detection, and global and local planning to provide a robust, reliable navigation capability. The navigation performance is such that we regularly allow the `PR2` to drive around our building unattended. The details of the navigation system are outside the scope of the present discussion, and will be presented in a future paper.

As shown in Figure 2, the map is manually annotated with the approximate locations of outlets and doors, the sides of the door handles (left/right), and the opening direction

[2]We scope our effort to institutional environments that conform to modern U.S. building codes, chief among them the Americans with Disabilities Act (ADA), which establishes a variety of constraints, including minimum width on doorways, and the type and placement of door handles.

[3]`ROS` (Robot Operating System - `http://ros.org`)

[4]`OpenCV` (Computer Vision Library - Open Computer Vision - `http://opencv.willowgarage.com`)

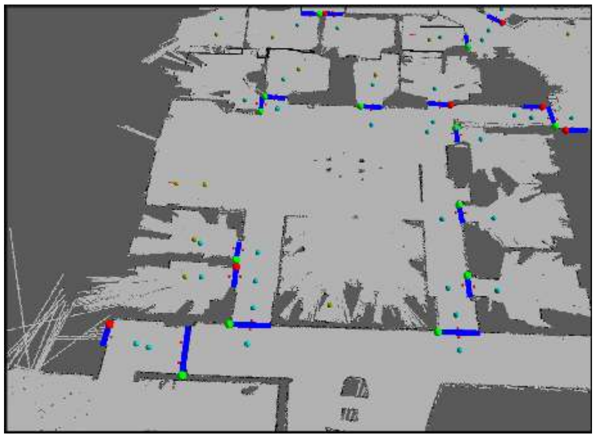


Fig. 2. The 2D map of the office. Light grey is open space, and dark grey marks obstacles and unknown space. Blue lines mark the doorways with a large green or red dot marking the hinge. Green hinges open counter-clockwise, red hinges open clockwise. Blue dots show the goal points that the robot approaches when it needs to interact with the corresponding door.

of doors (clockwise/counter-clockwise). In a more complete system, these could be found automatically.

b) Executive control: TREX [21] [22], a model-based hybrid executive, integrates navigation, door-opening, and plugging-in capabilities to robustly achieve high-level goals (e.g. recharge at outlet nr. 6). TREX combines task planning and execution in a unified framework for real-world domains with discrete and continuous states, durative and concurrent actions, and deliberative and reactive behavior.

Actions are the building blocks of TREX programs. An action is a modular, goal-achieving behavior implemented as an independent `ROS` node. An action is *active* when tasked to achieve a goal. Otherwise it is *inactive*, operating at a very low duty cycle. TREX offers a range of capabilities pertinent to robust action assembly and execution:

- Automated online **planning**. Execution is driven by a set of user-selected, unordered, high-level goals. These goals are scheduled and transformed into action sequences through automated planning.
- Concise specifications for **state machines**. Robust execution strategies can be encoded as state machines, providing a high-level programming model, and affording non-linear, reactive approaches to action selection. These state machines are directly integrated into the planning framework.
- Ability to express and enforce **configuration constraints**. For example, to avoid collision when driving around, the tilt laser must be running and the arms must be stowed. Actions often require specific real-time controllers: *untucking the arms* uses a joint-space trajectory controller, whereas *grasping a handle* uses an effort controller.
- Ability to express and enforce **timing constraints**. Timing constraints can be used to force preemption of actions that have not completed within an allotted time. Timing constraints also arise where actions require concurrent execution (e.g. moving the base while pushing the door), or where precedence rules apply (e.g. do not start pushing implies the robot has touched it).

IV. Approach

Our approach to the door opening and plugging tasks, and manipulation tasks in general, uses a methodology for incorporating robustness and trading off error in perception and manipulation. The main ideas can be summarized as follows:

1) Analyze the perception and control requirements for the tasks, especially with regards to uncertainty in perception, calibration, and control.
2) Make a realistic error budget that allows for robust algorithms. For example, in the plugging task, the distance to the sockets can only be estimated to within about 1cm using template matching, because the appearance of the sockets can vary widely.
3) Utilize the strength of compliant control to compensate for uncertainty. For both doors and plugging, we develop control algorithms that are robust to dislocations within the error budget.
4) Don't rely on high-accuracy calibration throughout the robot. Calibration along long kinematic chains and several different sensors is difficult to achieve and maintain, and makes the system brittle. Instead, for the high-precision plugging task, we use *visual differencing* to find the pose of the plug and socket in a single camera frame, allowing precise relative control.
5) Design of action primitives with cognizant failure modes and explicit recovery strategies.

We now discuss the application of this design methodology to the tasks of opening doors and plugging into outlets.

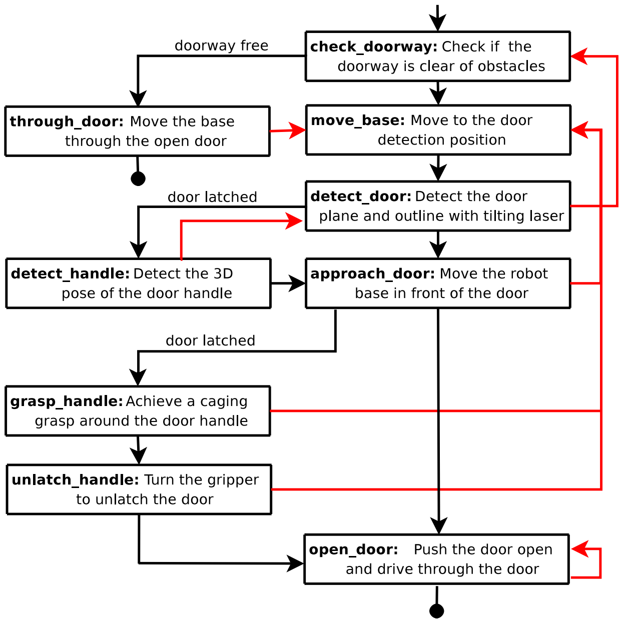

doorway free
check_doorway: Check if the doorway is clear of obstacles
through_door: Move the base through the open door
move_base: Move to the door detection position
door latched
detect_door: Detect the door plane and outline with tilting laser
detect_handle: Detect the 3D pose of the door handle
approach_door: Move the robot base in front of the door
door latched
grasp_handle: Achieve a caging grasp around the door handle
unlatch_handle: Turn the gripper to unlatch the door
open_door: Push the door open and drive through the door

Fig. 3. The state machine used by the door task. Black arrows show state transitions, red arrows show error recovery.

A. Door Task

We designed 14 door action primitives to complete the door task. Each door action performs one single task, such as detecting the door, grasping the door handle, moving the mobile base, etc. Figure 3 shows the state machine logic that stitches the individual actions together into a robust door detection and opening system. The black arrows show regular state transitions between actions, while the red arrows show the recovery logic in case an individual action fails. This section discusses the most important door actions in more detail.

1) Door Detection: The door detector operates directly on a 3D pointcloud acquired by the tilting laser scanner. Figure 4 shows a 3D pointcloud of a door and the surrounding environment. The displayed cloud density is obtained by a 10-second sweep of the tilting platform. The door detection algorithm is based on the segmentation and clustering approach described in [13] which generates a set of possible door candidates. For each door candidate we calculate a set of geometric attributes, such as width, height, area, etc, and eliminate those that do not comply with ADA (Americans with Disabilities Act) requirements. The remaining door candidates are ranked based on their distance from the expected door location in the 2D map.

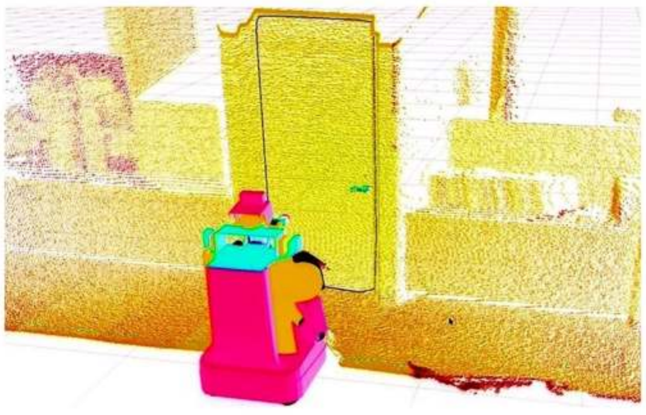


Fig. 4. Segmenting the door plane from a 3D point cloud acquired by the tilting laser scanner.

The door detection is based on geometric features, i.e. it detects the door plane and its boundaries. This means the detector requires the door plane to be distinguishable from the wall plane either because they are offset, there is an edge around the door, or the door is slightly open. When a door is completely flush with the wall – within the noise of the laser scanner – the detector will be unable to distinguish the door from the wall.

2) Handle Detection: Door handle detection is inherently difficult because the round metal handle introduces high inaccuracies in the distance measurements of the optical sensors, making it hard to distinguish the handle from the door plane. To achieve robust handle detection, without knowing the specific geometry of the doors used in the experiments, we employ two different detection methods in parallel: one that uses vision and stereo perception, and one that uses the laser scanner. Both methods need to agree on the handle location for a successful detection.

This combination of sensors with very different characteristics proved successful and completely eliminated false positives. However, it required re-trying logic to cope with failed detections. In order to avoid repeated failures that would occur when applying the detection algorithms on the same sensor measurements, every re-try uses new sensor measurements. After a failure, the robot is again commanded to its initial pose in front of the door, but because of sensor noise in navigation and localization, the robot ends up in a slighty different pose, from where new sensor measurements are abtained. This results in a 100% success rate within two cycles of five re-tries in our experiments.

Laser handle detection: The laser handle detection operates on the portion of the 3D pointcloud that lies within 0.1m from the door plane, and within the ADA height bounds. The segmentation of the handle points from the pointcloud is based on (i) a clustering approach that exploits the intensity differences between the handle and the door plane and (ii) an approach that takes into account differences in the local surface curvature near the door handle. Both approaches are described in [13].

Vision and stereo handle detection: The vision based handle detection uses the Viola and Jones object classifier cascade [23] as implemented by OpenCV's *HaarClassifier-Cascade* methods. The classifier is trained on a set of door handle images (we used over 200 in our experiments) that capture different viewing angles and changes in illumination of the door handles present in our building. We use a low detection threshold to always detect the handle, at the cost of a relatively high number of false positives. The 3D range information from the stereo camera is then used to filter out the false positives by checking whether 3D points are present at the detected location. Handles found at the wrong height, scale, or depth variation are rejected. Finally, we repeat the detection process across 7 consecutive frames and use spatial voting to filter out spurious false positives. Figure 5 shows a visualization of the handle detection process.

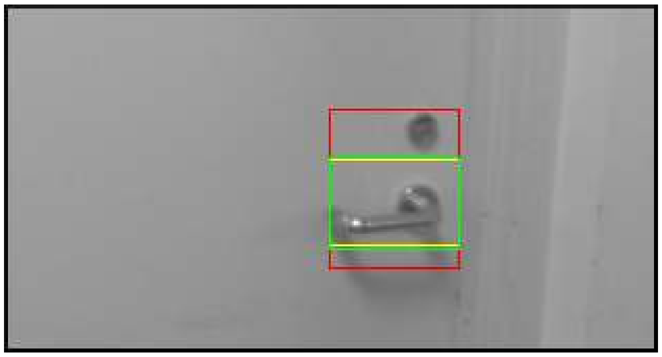


Fig. 5. Vision door handle detection. The green box is the final detection vote over seven frames, with intermediate detections represented by yellow boxes. The red box is a false detection filtered out by scale considerations.

3) Door Opening: The door opening subtask consists of approaching the door, grasping the door handle, unlatching the handle and pushing the door open while driving through. Each of these tasks uses one or more door actions, as shown in Figure 3.

Grasping the door handle: The robot approaches the door to within grasping distance, based on the detected 3D door pose. From the approach point, the arm reaches for the handle to achieve a caging grasp around the door handle. Note that the pose of the door handle is not re-detected after the robot approaches the door; rather, the system relies on accurate wheel odometry to transform the detected 3D pose of the door handle to the new local robot frame. The grasping motion is performed without new sensor feedback. Still, the robot reliably grasps the handle because the grasp tolerances are much higher than the noise in the perception or arm calibration errors.

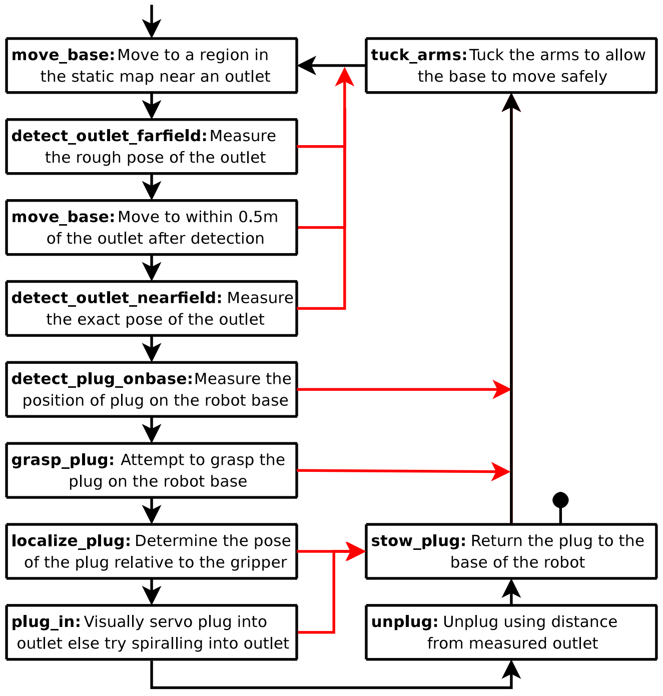
Compliant manipulation: With the end-effector holding on to the door handle, the motion degrees of freedom (DOF) of the arm are limited to the two DOF of the door handle: the rotation of the door around its hinges, and the unlatching motion of the handle. However, no kinematic model of the door is available because we attempt to use as little prior information as possible. To unlatch and open the door under these constraints, we employ the *Task Frame Formalism* (TFF) [24], which is an intuitive interface for compliant and force controlled tasks in the *Hybrid Control Paradigm* [25]. The task frame is rigidly attached to the center of the door handle, with its y-axis along the handle, and its x-axis normal to the door plane. The pose of the task frame is tracked over time based on the gripper pose. The translation along the x axis of the task frame and the rotation around the y axis are velocity controlled. The remaining four DOF of the end-effector are force controlled, making the arm compliant to the motion of the door. As the door opens, the force feedback loop senses the motion of the door, keeping the end-effector perpendicular to the door plane as the door opens, without requiring a geometric door model. To unlatch the handle, a torque of 3 Nm is applied.

Base-arm interaction: While only the robot arm is responsible for opening the door, the base moves to increase the workspace of the arm. The base continuously moves to within 5cm of the door, edging forward as the door opens more and more. To avoid collisions between the base and its environment, the base controller relies on measurements from the base laser to detect the door pose. The motion of the mobile base is specified independently of the arm motion; however, the base and the arm do affect each other through the environment. When the base moves, the arm senses and follows the reaction forces of the environment in the DOF constrained by the door. While the arm pushes the door open, the space between the base and the door increases, triggering the base to move forward.

B. Plugging Task

We designed eight plug action primitives to complete the plugging in task. Each plug action performs a single subtask, such as detecting the outlet, grasping the plug, moving the mobile base, etc. Figure 6 shows the state machine logic that stitches the individual actions together into a robust outlet detection and plugging system. The plug state logic requires the robot to be within a 3m x 3m area near an outlet before starting the plugging task.

The strategy is to roughly locate the outlet from a distance of up to 3m, move to a location near an outlet, and then perform precise pose estimation of both the socket and plug in the same camera frame. This *visual differencing* step obviates the need for precise mechanical calibration, which is difficult to achieve and maintain. This section describes in detail the key aspects of the plugging task.



move_base: Move to a region in the static map near an outlet
tuck_arms: Tuck the arms to allow the base to move safely
detect_outlet_farfield: Measure the rough pose of the outlet
move_base: Move to within 0.5m of the outlet after detection
detect_outlet_nearfield: Measure the exact pose of the outlet
detect_plug_onbase: Measure the position of plug on the robot base
grasp_plug: Attempt to grasp the plug on the robot base
localize_plug: Determine the pose of the plug relative to the gripper
stow_plug: Return the plug to the base of the robot
plug_in: Visually servo plug into outlet else try spiralling into outlet
unplug: Unplug using distance from measured outlet

Fig. 6. The sequence of actions that comprise the plug-in task. Black arrows show state transitions, red show error recovery.

1) Farfield Outlet Identification and Localization: To begin plugging in, the robot uses the *navigation stack* and the *topological map* to navigate to within 3 meters of the outlet. The *farfield outlet identification and localization* action is then used to detect the outlet. This action uses the stereo camera to extract a disparity image which encodes the offsets between matching texture patches in the left and right cameras. Outlets create a texture pattern that show up reliably in the disparity image. For outlet detection, all areas that have no disparity (blank walls for example) are ignored. The disparity image is then turned into a depth map and all outlet candidate locations that are non planar, too large or small, or too high or low, are rejected.

Next, the base laser scanner is used to find the pose of the walls, which is used to rectify the candidate outlet patches into frontal views by removing their perspective distortion. Template matching is then run on the frontal views of the outlet candidates (see figure 7), to eliminate the remaining false candidates. The computed poses of the remaining candidates are passed to the navigation stack, which drives the robot to within 0.5m of the closest outlet. During experimental testing of this action, the outlet detection was successful 84% of the time (44 of 52 trials). The 8 failures were then used to create better datasets which resulted in a 100% success rate during fully integrated trials.

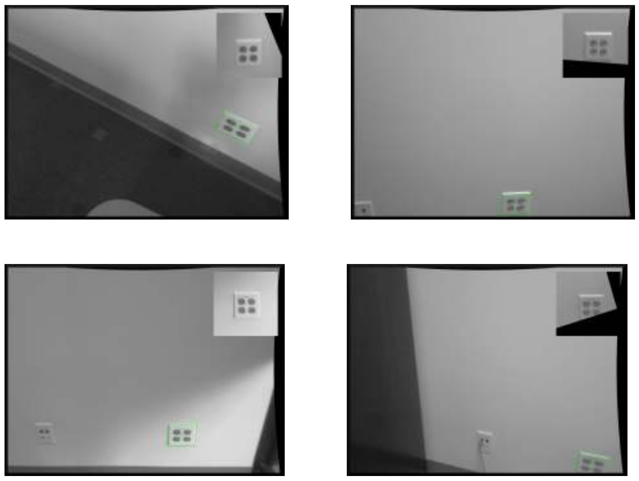


Fig. 7. Outlet detections with varying pose, illumination, and partial occlusion. The resulting rectified outlet patch is also shown.

2) Nearfield Outlet Localization: Nearfield outlet detection determines the precise pose of the outlet. This action primitive requires the robot's base to be within 0.5m of the outlet. The key challenges are camera angle (60 degrees) and distance from the outlet (∼1.5m). From this viewpoint, the ground hole on the outlet is approximately seven pixels in diameter using a 5MP Prosilica camera with a 16mm lens. The outlet holes tend to be low contrast with limited resolution and high perspective distortion.

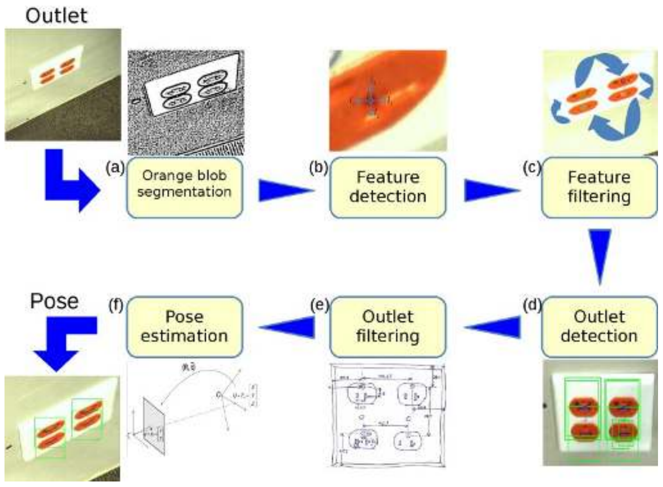


Fig. 8. Processing flow chart for finding outlet pose.

In experiments we used only the 2x2 orange on white outlets readily available in our building. Figure 8 depicts the outlet detection algorithm: (a) We adaptively threshold the grayscale image and look for four socket-sized components grouped together. If unsuccessful, we morphologically grow the thresholded image to connect outlet regions potentially split by lighting or perspective effects. (b) Within each of the connected components, we search for smaller connected components (potential outlet holes). We measure the contrast of the holes by comparing the central pixel intensity I_c to four surrounding pixels offset by 1.5 times the diameter of the small component. Contrast is defined as $c = 1/(4I_c)\sum_{k=1..4} I_k$. We accept a hole if $c > 1.1$. (c) We disregard features outside of the outlet blobs. (d) We group nearby hole features into potential outlet configurations. There may be multiple hypotheses within each connected component. (e) We compute a homography from the centers of the four outlet blobs and use it to produce a straight-on rectified view of the outlet. Rectified hole configurations are verified against an outlet measurement model to reject configurations with distance between slot holes differing more than 20% from the expected distance (12mm). (f) If we have a confirmed outlet region, we solve the planar PnP problem, with outlet hole locations and measurement model as inputs, to find the pose of the outlet relative to the camera. This approach achieved a 100% recognition rate on the dataset of 87 outlet test images that we collected. Our technique was fairly specific to the 2x2 orange on white outlets in our building. We have since switched to the much more general technique of using geometric hashing to match a learned outlet template to the outlet being observed.

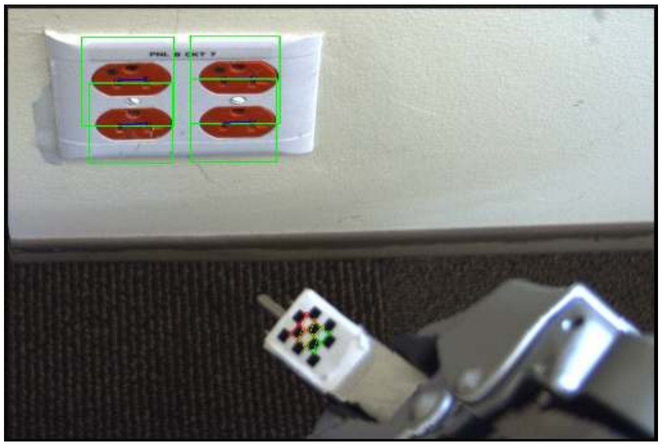


Fig. 9. Plug localization using a checkerboard. Localizing the outlet and plug in the same camera frame allows us to use visual servoing.

3) Grasping and Localizing the Plug in the Gripper: The plug was attached to the base of the PR2 using strong earth magnets and was placed such that it was only possible to determine the plugs location using the tilting laser scanner. The step of remeasuring the plug was necessary during the experiments because the plug would shift around and two robots were used with different magnet locations. Plug detection on the base of the robot is done directly on the 3D pointcloud from the tilting laser. The centroid of the plug is detected from a pointcloud collected from a 3-second, 6-degree motion of the tilting laser. This detection method estimates the plane of the base cover, then clusters all points above the base plane and returns the centroid of the candidate plug cluster. The centroid of the cluster is averaged for accuracy over 4 positive detections and then used to grasp the plug from the base of the robot. The physical dimensions of the plug are initially used to determine that the plug is retrieved from the base. The plug localization step is later used to judge the quality of the grasp and helps determine if the plug should be stowed and regrasped to improve the grasp quality. The detection of the plug on the base during experimental studies was 100% (48 of 48 trials), grasping the plug is 91% (44 of 48 trials). Failure to grasp the plug from the base of the robot was due to calibration errors, which were handled by redetecting the plug and attempting the grasp again. In the four trials that missed grasping the plug, the second attempt succeeded.

To track the precise pose of the plug in the gripper, we again use the 5MP Prosilica camera. We installed a small checkerboard on the plug attached to `PR2`. Engineering the checkerboard to fit on the plug involves tradeoffs. Too few detectable interior corners may make the pose estimation problem ill-conditioned, with multiple pose estimates fitting the data about equally well. On the other hand, squares that are too small result in high relative error in the detected corner positions. We used a 5x6 (4x5=20 interior corners) checkerboard with 4.2mm squares. Robustness is improved by using the pose of the end-effector in the camera frame as the initial estimate for the plug pose. We gain a speedup by performing detection on a small region-of-interest within each 5MP frame surrounding the projected position of the end-effector. To comfirm the absolute accuracy of the detector, the plug was moved along a high-precision caliper rig, showing an accuracy of ± 0.5mm, and ± 3 degrees in yaw. Additionally, localizing the plug in the gripper was successful 100% of the time (32 of 32 trials).

4) Plugging In: The key challenge for plugging in resided in the extremely small error budget and the difficulty in calibrating the robot to the accuracy needed. For success the center of the ground post of the plug cannot be greater than 2mm from the center of the ground hole in the outlet socket. The `PR2`, however, is only able to maintain a camera to end effector accuracy of approximately 2cm during an open loop manipulation task. By using relative pose estimates of the plug and socket in the camera frame (visual differencing), only a rough calibration is needed to move the arm in a direction that reduces the difference.

The error contribution from the plug is 0.5mm, and from the socket it is about 0.5mm in directions perpendicular to the pointing axis of the camera. Unfortunately, the depth error along the axis is on the order of 1cm, based on deviations from the socket template. Direct application of visual differencing to place the plug was successful approximately 10% of the time (2 of 19 trials), but still reduced the search space for plugging in by an order of magnitude over calibration. After attempting direct plugging and not succeeding, robot behavior defaults to spiralling outward from the best guess in 2mm increments until timing out or succeeding. In experimental trials, this alternative brute force method, in combination with visual servoing, was successful 95% of the time (18 of 19 trials).

V. Integrated Experimental Results

Our primary experimental goal was to verify that the fully integrated system could robustly open doors, navigate the building, and plug itself in. The robustness of our system was demonstrated on 2 occasions where multiple high-level recharge goals were achieved in a busy office environment without external intervention. The striking result of these experiments is that despite repeated failure of a number of individual components, the overall system was robust enough to adequately recover, achieving all feasible goals.

The trials were conducted with 10 and 9 outlet goals respectively. A video of the first trial is included with this paper. In the first trial, one office was locked and remained so throughout. In the second trial, four offices were locked, with one of those locked doors becoming unlocked while the trial was running. Table I details execution statistics of selected actions.

TABLE I
Results from two independent, fully integrated trial runs.

Trial 1: 10 outlet goals, 1 behind locked door, 1 behind open door
Trial 2: 9 outlet goals, 3 behind locked doors, 1 behindopen door

	success		**abort**		**preempt**	
Door Actions	T1	T2	T1	T2	T1	T2
detect door	9	8	5	0	0	0
detect handle	4	5	3	3	0	0
grasp handle	4	5	0	0	0	0
unlatch handle	3	3	0	0	0	0
open door	0	0	0	0	3	3
release handle	4	4	0	0	0	0
touch door	5	5	0	0	0	0
push door	0	0	0	0	5	5
Plug Actions						
detect outlet coarse	11	20	0	0	0	0
detect outlet fine	11	7	0	0	0	0
detect plug on base	11	7	0	0	0	0
grasp plug	9	7	2	0	0	0
localize plug	9	7	0	0	0	0
plug in	9	6	0	0	0	1
unplug	9	6	0	0	0	0
stow plug	9	6	0	0	0	0
Navigation Actions						
move base	75	86	4	23	0	0
check doorway	26	20	0	0	0	0

The data for the *plug_in* action show that in Trial 1 `PR2` plugged in 9 times while in Trial 2 this occurred only 6 times, reflecting the number of locked doors. The *push_door* and *open_door* actions were always explicitly preempted by the executive as part of a nominal behavior by executing a *stop* action. This is because these actions do not have the context in which to decide when they are complete. Rather, completion is determined based on the termination of the *move_base_door* action.

In some cases, a timeout was usefully applied. For example, in Trial 2, the *plug_in* action was preempted on one occasion. The state machine required `PR2` to reposition at the initial pose and retry, which succeeded.

Some actions were prone to failure. Local and global navigation both suffered from improperly detected obstacles arising from laser hits on the robot shoulder. Even with the arm tucked, the shoulder protruded slightly from the robot footprint, which was not filtered correctly. Door and handle detection also failed a number of times, despite internally encoded retry logic. In the case of handle detection, it was often sufficient simply to re-detect the door, providing new input for handle detection. Door detection fails when presented with an open doorway because the laser cannot see the door plane. This can occur because checking the path may erroneously declare the doorway is not clear. Repeated failure to detect an open door results in temporarily giving up that goal; when the goal is revisited at a later time, noise in navigation and localization would result in a slightly different detection pose.

Observe that *unlatch_handle* always succeeded in the first trial, despite the door being locked. This is because the scenarios for success include determining that the door is locked. Also note that this same action aborted 3 times in the second trial. In that run, the gripper calibration was slightly off and the *grasp_handle* action was tending to grasp too far to the side of the handle, allowing the grasp to slip when force was applied. Recovery by repositioning the robot and starting again proved an effective remedy.

When the robot navigates through the building, it crosses many open doorways. At each doorway, the check doorway action ensures the door is open before navigating through.

Finally, the second trial illustrated a recovery at the highest level where an initially locked door was later unlocked. The executive in this case deferred that goal and succeeded upon re-trial.

VI. Conclusions

The overall goal of this paper was to demonstrate a system capable of reliably navigating a building, opening doors, and plugging in as needed. These tasks were completed in an unaltered building environment without the use of a base or docking station. These claims were confirmed through two independent trials of the fully integrated system. This system demonstrated robustness on two levels. The action primitives used redundant sensor data in combination with compliant control of the manipulator to achieve successful detection and manipulation of doors and plugs. The executive used recovery behaviors and timeouts on action completion to ensure task level success across action primitives. Our approach used a generalized mobile manipulation platform, `PR2`, for navigation, door opening, and recharging, without specific design consideration for any particular task.

The results of this paper can be reproduced in simulation using ROS and Gazebo, an open-source multi-robot simulator [26]. For instructions visit the Willow Garage wiki at `http://pr.willowgarage.com/wiki/doors_and_plugs`.

VII. Acknowledgements

The authors of this paper wish to acknowledge the ROS and `PR2` development teams for working hard to create a stable robotic system and platform for completing our research. Also we would like to thank Ken Conley for film and editing the video of the experimental trials.

References

[1] S. Roh, J. Park, Y. Lee, Y. Song, K. Yang, M. Choi, H. Kim, H. Lee, and H. Choi, "Flexible Docking Mechanism with Error-Compensation Capability for Auto Recharging System of Mobile Robot," *Intl. J. of Control, Automation, and Systems*, vol. 6, pp. 731–739, Oct. 2008.

[2] M. Silverman, B. Jung, D. Nies, and G. Sukhatme, "Staying Alive Longer: Autonomous Robot Recharging Put to the Test," USC Center for Robotics and Embedded Systems, Tech. Rep. CRES-03-015, 2003.

[3] E. Torres-Jara, "A Self-Feeding Robot," Master's thesis, Massachusetts Institute of Technology, Jan. 2006.

[4] T. Yamada, K. Nagatani, and Y. Tanaka, "Autonomous Insertion of a Plug into Real Electrical Outlet by a Mobile Manipulator," in *Proc. of the Intl. Conf. on Field and Service Robotics (FSR)*, P. Corke and S. Sukkarieh, Eds. Springer, 2006, vol. 25, pp. 389–400.

[5] L. Bustamante and J. Gu, "Localization of Electrical Outlet for a Mobile Robot using Visual Servoing," in *Canadian Conf. on Electrical and Computer Engineering*, Apr. 2007, pp. 1211–1214.

[6] iRobot, "Roomba Cleaning Robot with Self-Charging Home Base Station," http://www.irobot.com.

[7] Sony, "AIBO Entertainment Robot with Self-Charging Function," http://support.sony-europe.com/aibo/index.asp.

[8] E. Aude, E. Lopes, C. Aguiar, and M. Martins, "Door Crossing and State Identification Using Robotic Vision," in *Intl. IFAC Symp. on Robot Control (SYROCO)*, 2006.

[9] A. Jain and C. C. Kemp, "Behaviors for Robust Door Opening and Doorway Traversal with a Force-Sensing Mobile Manipulator," in *RSS Manipulation Workshop: Intelligence in Human Environments*, 2008.

[10] A. Y. N. Ellen Klingbeil, Ashutosh Saxena, "Learning to Open New Doors," in *RSS Workshop on Robot Manipulation*, 2008.

[11] C. Ott, B. Baeuml, C. Borst, and G. Hirzinger, "Autonomous Opening of a Door with a Mobile Manipulator: A Case Study," in *Proc. of the IFAC Symp. on Intelligent Autonomous Vehicles (IAV)*, 2007.

[12] A. Jain and C. C. Kemp, "Behavior-Based Door Opening with Equilibrium Point Control," in *RSS Workshop on Mobile Manipulation*, 2009.

[13] R. B. Rusu, W. Meeussen, S. Chitta, and M. Beetz, "Laser-based perception for door and handle identification," in *Proc. of the Intl. Conf. on Advanced Robotics (ICAR)*, June 2009.

[14] K. Nagatani and S. Yuta, "An experiment on opening-door-behavior by an autonomous mobile robot with a manipulator," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 1995, pp. 45–50.

[15] G. Niemeyer and J. Slotine, "A simple strategy for opening an unknown door," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 2, 1997.

[16] L. Petersson, D. Austin, and D. Kragic, "High-level control of a mobile manipulator for door opening," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2000.

[17] C. Rhee, W. Chung, M. Kim, Y. Shim, and H. Lee, "Door opening control using the multi-fingered robotic hand for the indoor service robot," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2004.

[18] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on Open-Source Software*, 2009.

[19] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, Inc., 2008, pp. 415–453.

[20] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The office marathon: Robust navigation in an indoor office environment," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2010.

[21] C. McGann, F. Py, K. Rajan, H. Thomas, R. Henthorn, and R. McEwen, "A deliberative architecture for auv control," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2008, pp. 1049–1054.

[22] C. McGann, E. Berger, J. Bohren, S. Chitta, B. Gerkey, S. Glaser, B. Marthi, W. Meeussen, T. Pratkanis, E. Marder-Eppstein, and M. Wise, "Model-based, Hierarchical Control of a Mobile Manipulation Platform," in *ICAPS Workshop on Planning and Plan Execution for Real-World Systems*, Sept. 2009.

[23] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," *Proc. of Computer Vision and Pattern Recognition Conf. (CVPR)*, vol. 1, pp. 511–517, 2001.

[24] M. T. Mason, "Compliance and force control for computer controlled manipulators," *IEEE Transactions on Systems, Man, and Cybernetics (SMC)*, vol. 11, no. 6, pp. 418–432, 1981.

[25] M. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Transactions of the ASME, J. of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 126–133, 1981.

[26] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Sept. 2004, pp. 2149–2154.