Real-Time Self Collision Avoidance for Humanoids by means of Nullspace Criteria and Task Intervals

Hisashi Sugiura, Michael Gienger, Herbert Janssen, Christian Goerick
Honda Research Institute Europe GmbH
Carl-Legien Strasse 30
D-63073 Offenbach/Main, Germany
Email: hisashi.sugiura@honda-ri.de

Abstract— **We describe a new method for real-time collision avoidance for humanoid robots. Instead of explicit control or modification of given target commands our method influences the control system by means of a nullspace criteria and a task interval. The nullspace criteria is driven by a virtual force acting on a joint center vector that defines the minimum of a potential function in joint space. The task interval defines the target constraints in task coordinates and allows the avoidance system to specify deviations from the given target position. The advantages of this indirect method are that continuous motion generation is supported and the underlying motion control may use any trajectory generation method that is able to satisfy the constraints given by the collision avoidance. It is most useful for highly redundant robots like typical humanoids. The method is able to assure smooth collision free movement on the humanoid robot ASIMO in real time interaction even in cases where the dynamical constraints of legged walking apply.**

keywords: humanoid robot, collision avoidance, whole body motion

I. Introduction

For humanoid robots recently requirements such as autonomy, interactivity and robustness became more and more important. In the real world and especially in interaction with a human movement targets cannot be predicted in advance and planning methods generally become less attractive. Nevertheless, it is very important to detect and avoid self-collision and obstacles both by prediction and planning with reactivity. In order not to break the robot, some traditional systems freeze the robot in a so called "*emergency stop*" which means a discontinuous stop of all movements. But if dynamical constraints apply like e.g. in case of legged robots that have to keep dynamical balance, both the robot and its environment including interacting humans may be in danger.

The purpose of the work presented in this paper is real-time self-collision avoidance between the body and both arms of our humanoid robot ASIMO, i.e. allow any movement target to be given interactively without breaking the robot by collisions of its link segments and continuous robot's motions unless the target or at least a collision free posture close to the target is reached.

There have been many papers presented about collision avoidance. Collision avoidance methods can be roughly distinguished into two categories. The first type of methods uses information about all obstacles in advance. Potential field methods[1] are very popular in this category. The second type of methods is reactive and obstacle information is only considered under certain circumstances. This type of method is attractive because it requires less computation time and is less dependent on the overall complexity of the scene[2], [3], [4], [7], [10].

The collision avoidance method we propose here is in the latter category since we want to deal with highly dynamic environments and for safety reasons use only onboard computation resources.

II. Distance computation

For our collision avoidance method we need to compute the closest points and thus the distance between the *segments* of the robot which are the physical links separated by joints[9]. All segments are modeled by *SSLs* (sphere swept line) or spheres since computation based on the real shape of the segments is impossible within our real-time and hardware constraints[6]. The segment model of the robot is shown in Fig.1. The closest points and distances are computed between all possible segment pairs.

Since the computational complexity of pairwise distance computation is $O(N^2)$ the robot's collision model has two layers. One is a coarse model consisting of one primitive per segment and the other is a fine model which can use up to 10 primitives in case of the upper body. All segment pairs are first computed based on the coarse model. The fine model is only used when a pair of coarse model primitives is close enough to each other *(warning zone)*.

III. Collision avoidance

We define the requirements for collision avoidance as follows.

- The segments should not come closer to each other than a certain minimum distance *(red zone)*, otherwise the robot will freeze and all (arm and upper body) motion will be stopped. This is an emergency situation and should never happen during normal operation.
- If segments come closer to each other than another certain distance *(yellow zone)* they should be "pushed away" from each other.
- The motion should not be terminated unless the target or at least a position close to the target is reached.

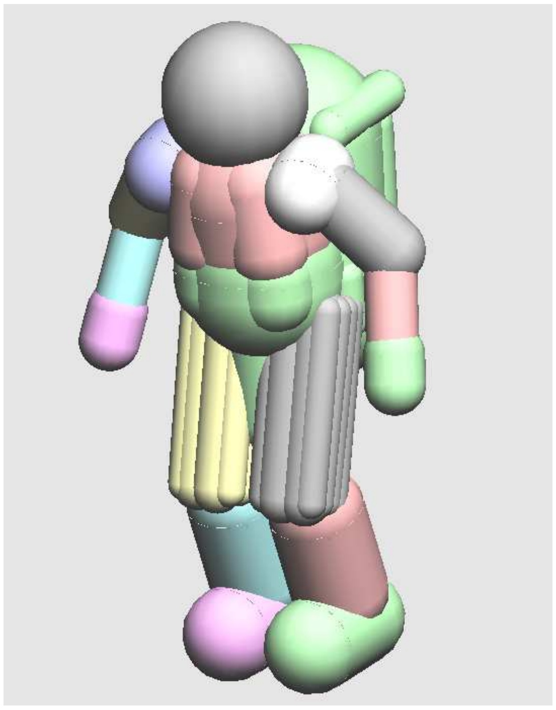


Fig. 1. Asimo's collision model is composed of sphere-swept-lines and spheres. Differing colors were used for visibility.

- A target should always be reached exactly if at all possible, i.e. if the target is outside the yellow zone.

A. Whole Body Motion Control

To control a robot with redundant degrees of freedom, we use a whole body motion control system with nullspace optimization criteria. This means that the task space target is always followed while the nullspace criteria is only used to derive a unique solution for the redundant kinematics.

In this control method, each joint velocity is computed as

$$\dot{\mathbf{q}} = \mathbf{J}^{\#}\dot{\mathbf{x}}_{task} + \mathbf{N}\boldsymbol{\xi} \tag{1}$$

$$\mathbf{N} = \mathbf{I} - \mathbf{J}^{\#}\mathbf{J} \tag{2}$$

where $\mathbf{N}$ is a linear matrix which maps an arbitarary joint velocity vector $\boldsymbol{\xi}$ into the nullspace and $\dot{\mathbf{q}}$ is the joint velocity, $\mathbf{J}^{\#}$ the pseudo inverse Jacobian, $\dot{\mathbf{x}}_{task}$ the target command in task space, $\mathbf{I}$ the identity matrix.

The nullspace optimization criteria can be expressed as a cost function $g(\mathbf{q})$ with

$$\dot{\mathbf{q}} = \mathbf{J}^{\#}\dot{\mathbf{x}}_{task} - \mathbf{N}(\frac{\partial g}{\partial \mathbf{q}})^T \quad . \tag{3}$$

In our case we chose the cost function to penalize deviations from an arbitrary joint center vector $\tilde{\mathbf{q}}$ by using a weighting matrix $\mathbf{W}$ as

$$g(\mathbf{q}) = \frac{1}{2}(\mathbf{q} - \tilde{\mathbf{q}})^T\mathbf{W}(\mathbf{q} - \tilde{\mathbf{q}}) \quad . \tag{4}$$

In the easiest case by choosing $\tilde{\mathbf{q}}$ accordingly, this cost function allows to keep away from the joint limits [5]. Additionally by using $\tilde{\mathbf{q}}$ as an input parameter to our control system we can easily shift the nullspace potential to prefer any externally given posture while simultaneously obeying the task space target velocity.

B. Virtual Force

To control the joint center vector we use a virtual force which is generated by virtual springs on all joints. They generate a force $\mathbf{f}$ proportional to the distance a joint moves from its center.

The virtual work principle is

$$\boldsymbol{\tau}^T\delta\mathbf{q} = \mathbf{f}^T\delta\mathbf{x} \tag{5}$$

where $\boldsymbol{\tau}$ is the joint torque vector, $\delta\mathbf{q}$ is the joint's angular increment vector and $\delta\mathbf{x}$ is a cartesian position increment vector. By means of the Jacobian transform $\delta\mathbf{x} = \mathbf{J}\delta\mathbf{q}$ we get

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{f} \quad . \tag{6}$$

This equation relates forces in cartesian space and torques in joint space.

In the easiest case we can control forces and torques by means of virtual springs. Thus the torque $\boldsymbol{\tau}_{virtual}$ is given by

$$\boldsymbol{\tau}_{virtual} = \mathbf{C}\Delta\mathbf{q} \tag{7}$$

where $\mathbf{C}$ is a compliance matrix for the virtual springs of the joints and the force $\mathbf{f}_{virtual}$ is given as

$$\mathbf{f}_{virtual} = k\Delta\mathbf{x} \tag{8}$$

where k is the spring constant.

In our collision avoidance method we call $\mathbf{p}$ the smallest of all model pair closest point vectors. If the length of this vector $\mathbf{p}$ is smaller than the yellow zone distance d_{yz} a non-zero $\Delta\mathbf{x}$ is the result.

$$\Delta\mathbf{x} = \begin{cases} 0 & \text{if} \quad |\mathbf{p}| > d_{yz} \\ (d_{yz}\frac{1}{|\mathbf{p}|} - 1)\mathbf{p} & \text{else} \end{cases} \tag{9}$$

Accordingly, $\Delta\mathbf{q}$ is used to describe the deviation between the actual joint vector and the joint center vector $\tilde{\mathbf{q}}$, $\mathbf{J}^T$ is identified as the transpose of the Jacobian of $\mathbf{p}$. This Jacobian is used for position control by the distance computation. e.g. control of the hand segment position with respect to the body.

Then we get following equation for $\Delta\mathbf{q}$:

$$\Delta\mathbf{q} = \mathbf{C}^{-1}\mathbf{J}^Tk\Delta\mathbf{x} \tag{10}$$

Finally we control the joint center vector $\tilde{\mathbf{q}}$ by moving it according to $\Delta\mathbf{q}$. In discrete formulation this can be written as

$$\tilde{\mathbf{q}}_{t+1} = \tilde{\mathbf{q}}_t + \Delta\mathbf{q} \quad . \tag{11}$$

This equation converts deviations of positions to deviations of joints without singularities. Thus the method is able to change the posture of the robot by applying a virtual force in task space and transforming it to a movement in joint space.

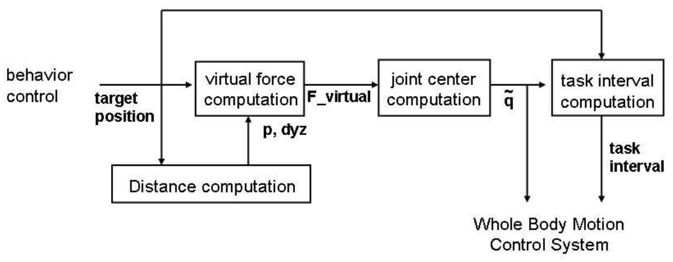


Fig. 2. Architecture of the collision avoidance and its interaction with the underlying whole body motion control system.

C. Task Interval

For many robot tasks, movement targets do not necessarily need to be specific cartesian points and orientations, e.g. for dancing or making gestures. Thus for some tasks, it is useful to specify the robot's movement targets as volumes in space, orientation intervals or generally speaking task intervals [8]. As far as the respective segment of the robot stays within the specified interval, the robot follows the nullspace criteria. If the task interval is zero, the target is a point and the robot simply moves to the target point. If the task interval is unlimited, then the robot follows the nullspace optimization criteria.

The advantage of the collision avoidance using these methods is that the target command is not modified explicitly, but it indirectly influences the control system by means of the joint center and the task interval.

Fig.3 exemplifies the system on Asimo. Fig.3(a) shows the robot in a resting position which is determined exclusively by the joint center since there is no target and thus no task space. In Fig.3(b) both arms are given targets and while the right arm has reached its target the left arm did not because an arm segment violates the yellow zone and the collision avoidance is activated. Fig.3(c) is a closeup of Fig.3(b). The forearm violates the yellow zone and a virtual force is generated based on the smallest closest point vector $\mathbf{p}$. This force pushes the arm back to the limit of the task interval. Note that the target position is not modified but the actual position stays within the task interval. If the target position moves out of the yellow zone, the task interval becomes zero and the arm moves to the target position accordingly.

IV. System

ASIMO is a humanoid robot which has 5 degrees of freedom for each arm, 2 degrees of freedom for the head and 6 degrees of freedom for each leg. Our whole body motion control system uses 21 degrees of freedom including virtual degrees of freedom that describe the robot's legs. The system of collision avoidance and its interaction with the control system is shown in Fig.2. Distance computation and avoidance are computed on ASIMO's internal computers because they are fundamental safety functions. Arm trajectories are generated by low pass filtering the difference between the target and the actual position. The task interval output from the collision avoidance is also low pass filtered before it is used in the control system to avoid jumping task space conditions.

Step generation for biped walking is switched on and off based on the deviation between actual position and target trajectory. If this deviation exceeds a threshold - in case of our experiments 50mm - the walking is automatically enabled.

V. Results

We tested our method by using both kinematics-only and dynamical simulations of Asimo. We ran extensive tests with over 100 randomly generated arm targets in our dynamics simulator but found no cases of red zone violation (emergency stop) or targets that could not be reached.

In our experiments we set the zones as follows:

- The *warning zone* distance is set to 100mm; entering it switches the distance computation from coarse to fine models.
- The *yellow zone* distance is set to 50mm; entering it activates collision avoidance.
- The *red zone* distance is set to 3mm; entering it triggers the emergency stop.

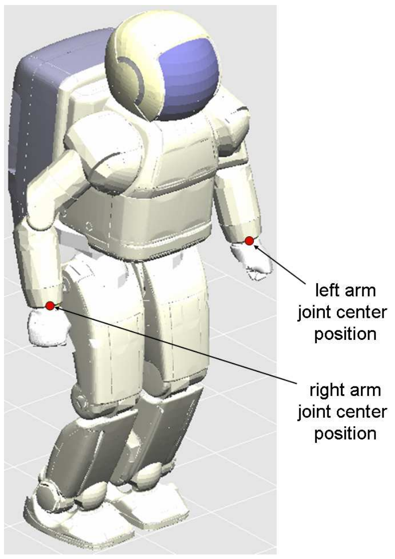
We will discuss two typical cases of our experiments below. Fig.4 shows an example of a left hand trajectory crossing the yellow zone. The walking is not activated in this example because the walking threshold is not exceeded. The start position and the target position are outside of the yellow zone, but time t1 to t2, the arm is inside of the yellow zone as shown in Fig.4(a). Therefore virtual force depicted in the center of Fig.4(b) is generated between t1 and t2. The task interval is depicted in the bottom of Fig.4(b). It is also generated from t1 but after t2 it does not immediately become zero because of the filtering discussed above. At time t3, the closest point model pair is switched as shown in top of Fig.4(b). At time t4, the actual position reaches the limit of the task interval as can be seen in the bottom of Fig.4(a). From now the arm moves directly to the target. The arm trajectory as modified by the collision avoidance. The arm moves to the final target even if it is currently inside of the yellow zone and finally reaches the target without stopping.

A second example - this time including walking - is given in Fig.5.

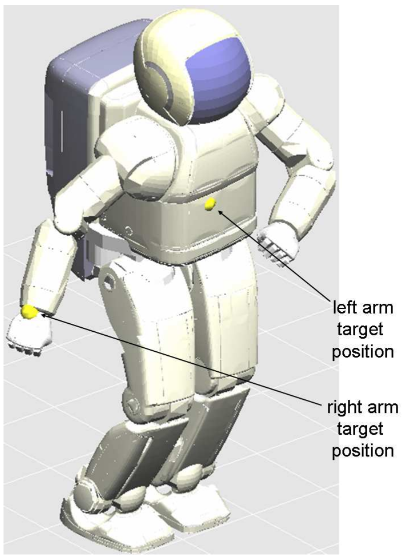
The yellow spheres denotes the given target positions. The image sequence shows the following steps:

(a) The target for the left arm is set. It is inside of the yellow zone.
(b) The robot tries to reach the target but fails, because the arm violates the yellow zone of the body. Thus the robot starts walking when the deviation threshold is reached.
(c) The robot continues to reach for the target. The yellow sphere close to the right hand is the target for the right arm.
(d) The robot stops walking and finally reaches the target.

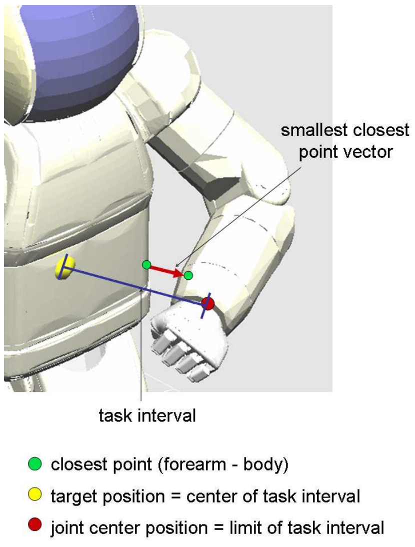
In this example the collision avoidance initally prohibits reaching the target but by means of causing a difference between the actual position and trajectory indirectly triggers walking so that the target can finally be reached.


left arm joint center position
right arm joint center position

(a) Resting position

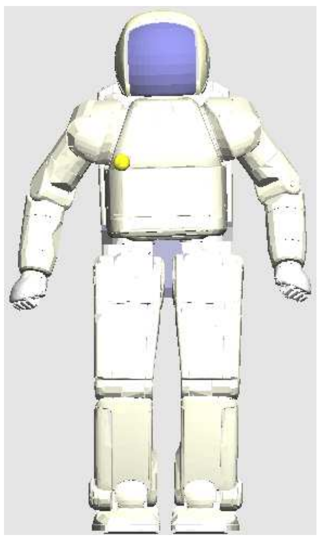


(b) Collision avoidance active

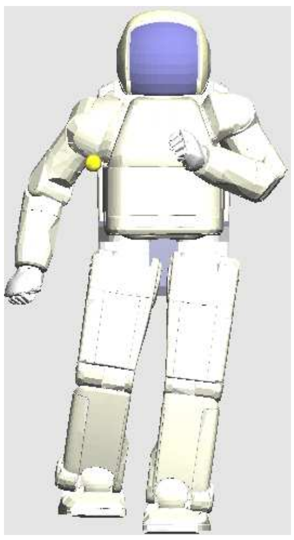

smallest closest point vector
task interval
closest point (forearm - body)
target position = center of task interval
joint center position = limit of task interval

(c) Closeup

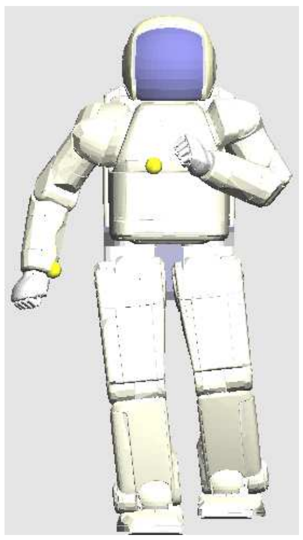
Fig. 3. Asimo and its control and avoidance system in some typical equilibrium states. (a) resting position, no targets given, only the default nullspace criteria apply. (b) the left arm target is inside the yellow zone, but virtual force and task interval ensure that the hand does not hit the upper leg. (c) closeup of (b), the blue line indicates the relevant part of the task interval



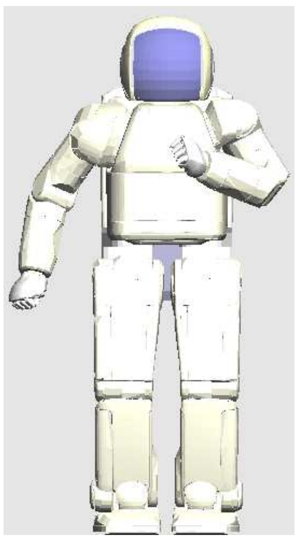
(a) Start: resting



(b) First step

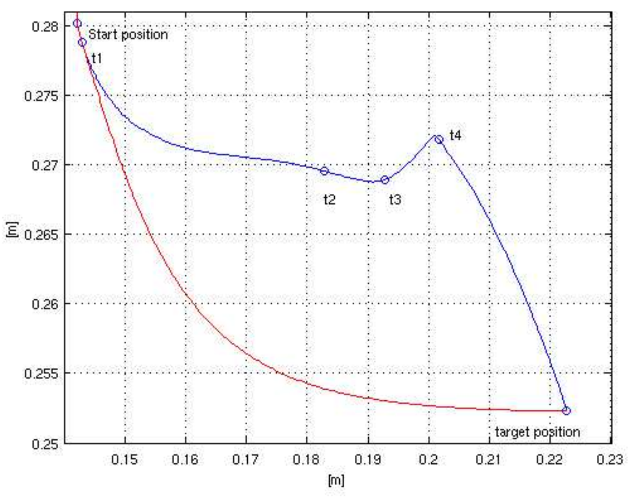


(c) Second step

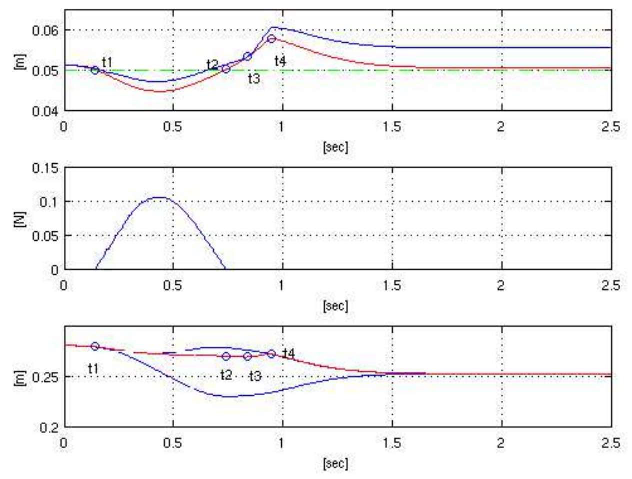


(d) Stop: target reached

Fig. 5. Simple example of a target that is inside the body and requires walking. The yellow spheres are the target positions.



(a) Trajectory in XY-plane



(b) Top: closest distance, Center: virtual force, Bottom: task interval

Fig. 4. Typical example of a trajectory crossing the yellow zone. t1 is the time of entering the yellow zone, t2 is the time of leaving the yellow zone, t3 is the time of switching between the closest point model pairs and t4 is time of the actual wrist position reaching the limits of the task interval. (a) XY-Plot of the left wrist position with (blue) and without (red) collision avoidance. (b) top: distance between closest points of the two closest model pairs hand - body1 (blue) and hand - body2 (red). (b) center: virtual force (b) bottom: actual Y position of the left wrist (red) and task interval borders (blue)

VI. Conclusion and Outlook

We described a collision avoidance method which uses virtual forces and task intervals. According to the results of our simulations, our robot moves smoothly without colliding when given various targets interactively.

We are currently working on extending the system to include external objects as detected by the vision system as obstacles.

Also we will adjust the robot segment model in real-time to increase the precision of the distance computation without increasing the computational load.

References

[1] Oussama Khatib *Real-Time Obstacle Avoidance for Manipulations and Mobile Robots* The international Journal of Robotics Research, Vol.5, No.1, pp. 90-98.,1986

[2] Anthony A. Maciejewski and Charles A. Klein *Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments* The international Journal of Robotics Research, Vol.4, No.3, pp. 109-117.,Fall 1985

[3] Oliver Brock, Oussama Khatib, Sriram Viji *Task Consistent Obstacle Avoidance and motion Behavior for Mobile Manipulation* In proceedings of the IEEE International Conference of Robotics and Automation, Washingon DC, USA May 2002

[4] Yoshihiko Nakamura *Advanced Robotics: Redundancy and Optimization.* Addison Wesley Publishing

[5] Michael Gienger, Herbert Janssen and Christian Goerick. *Task-Oriented Whole Body Motion for Humanoid Robots* In proceedings of the IEEE-RAS International Conference on Humanoid Robots, Tsukuba, Japan, December 2005

[6] Christer Ericson. *Real-time collision detection* The Morgan Kaufman Publishers

[7] Fumi Seto, Kazuhiro Kosuge, Yasuhisa Hirata. *Self-collision Avoidance Motion Control for Human Robot Cooperation System using RoBE* In proceedings of the IEEE/RSJ Internatiional Conference on Intelligent Robots and Systems, Edmonton, Canada, August 2005

[8] Michael Gienger, Herbert Janssen, Christian Goerick. *Exploiting Task Intervals for Whole Body Robot Control* Submitted to the IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing China, October 2006

[9] James Kuffner, Koichi Nishiwaki, Satoshi Kagami, Yasuo Kuniyoshi, Masayuki Inaba, Hirochika Inoue *Self-Collision Detection and Prevention for Humanoid Robots* In proceedings of the IEEE International Conference on Robotics and Automation, Wachington DC, USA, May 2002

[10] Ioannis Iossifidis, Gregor Schoener *Autonomous reaching and obstacle avoidance with the anthropomorphic arm of a robotic assistant using the attractor dynamics approach* In proceedings of the IEEE International Conference on Robotics and Automation, New Orleans, USA, April 2004