# $iCO_2$: a Networked Game for Collecting Large-Scale Eco-Driving Behavior Data

The $iCO_2$ game collects human driving behavior data at scale and better illuminates user behaviors, including which eco-driving practices drivers apply. The game uses the Distributed Virtual Environments (DiVE) networking framework for multiuser 3D virtual world applications. Tests of DiVE's "area of interest" method showed promising results on two important metrics: bandwidth and frames per second. A small marketing campaign also collected data from users interacting with the iPad version of $iCO_2$ in the wild.

**Helmut Prendinger**
*National Institute of Informatics, Tokyo*

**João Oliveira and João Catarino**
*Instituto Superior Técnico, Universidade de Lisboa*

**Marconi Madruga**
*National Institute of Informatics, Tokyo*

**Rui Prada**
*INESC-ID*

"Serious games" are digital games used for purposes other than mere entertainment. Those purposes — such as training, advertising, simulation, and education — offer potential benefits in many areas, including the military, education, corporate, and healthcare sectors.[1] Serious games let players experience situations that are difficult to reproduce in the real world due to issues such as safety, cost, and time. We believe that serious games can also positively affect players' development of practical skills, such as eco-safe driving.

Massively multiuser online games,[2] particularly on mobile platforms, have grown substantially in recent years.[3] One reason for the huge popularity of such networked environments is the fun of social communication, which is enabled by players' real-time interaction with graphical representations (avatars) of real people, including their friends.[4]

In our lab, we developed $iCO_2$ ("my $CO_2$"), a networked serious game that supports players in applying eco-friendly techniques when driving a car. The design and development of $iCO_2$ was influenced by two key factors. First, we wanted to ensure that researchers could use it to collect human driving behavior data at a large scale.[5] In particular, traffic engineers can use $iCO_2$ as a platform to collect rich data on human drivers' eco-driving techniques. The platform can also help test the effect of innovative intelligent transport systems.[6] Second, we wanted to make $iCO_2$ a fun and engaging game so that users would want to play it. The game's networking feature is important for both perspectives: in terms of research, users are expected

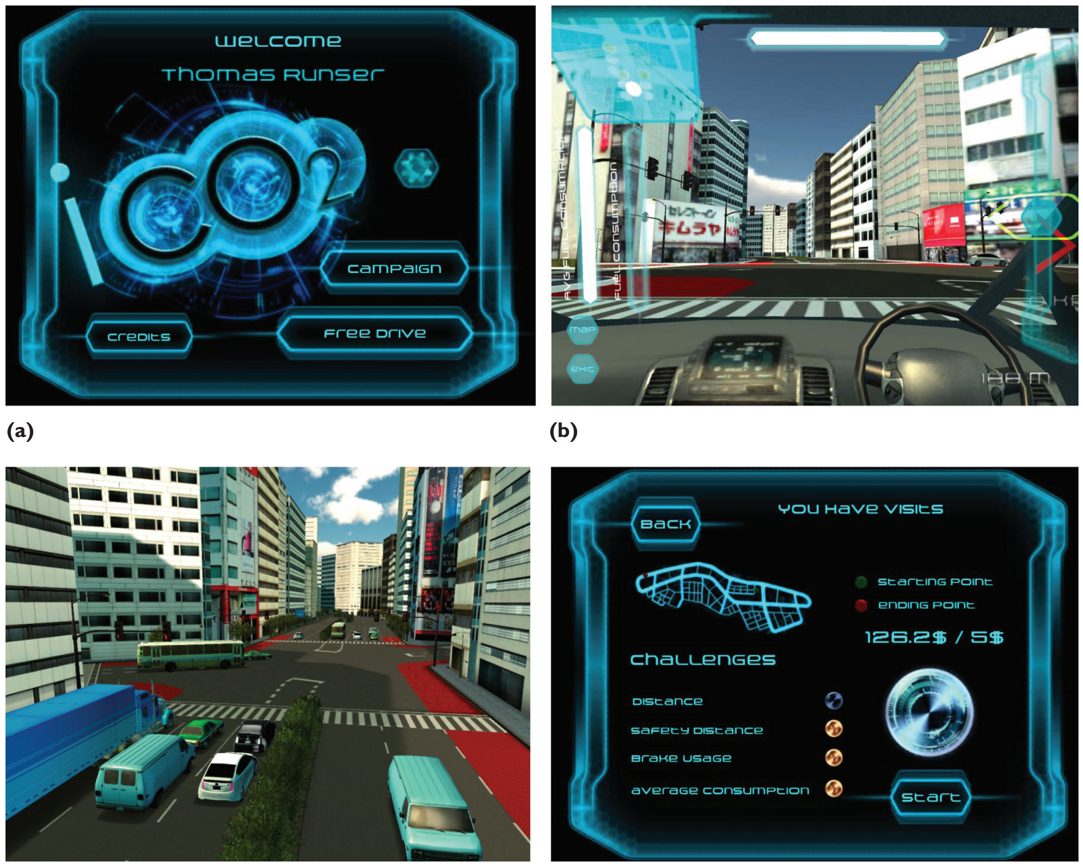

*Figure 1. The $iCO_2$ game. (a) The login interface. (b) A screen shot from an iPad. (c) A "bird's view" on virtual Tokyo. (d) An interface showing the player's progress.*

to drive more naturally among other (networked) human drivers (versus computer-controlled vehicles); also, for users, driving with other people makes the game more socially engaging.

## The $iCO_2$ Game

The game is situated in a replica of a 1-square kilometer area of Tokyo. The streets are populated with non-player character (NPC) cars driving on the roads. New and unexpected traffic situations are constantly being created as players interact with each other and with the NPC cars, as in the real world (see Figure 1c). Users can currently play the game on the Web (via mouse device input) and the iPad (via touch/tilt). Figure 1a shows the login screen and Figure 1b shows a screen shot from the iPad version. Players from both platforms can see each other in the shared environment. We developed $iCO_2$ in Unity3D (www.unity3d.com). For comparison, see Francesco Bella's work, which describes a standard use of a driving simulator.[7]

The networked game has two different game modes that share the same game environment — that is, users in either mode can see each other. In "Free Drive" mode, the goal is to drive as far as possible with the given amount of fuel. The incentive here is to be the most eco-friendly driver; players can compare their performance with that of other players using the score ranking. To achieve a good score, players must drive in an eco-friendly way, by accelerating and decelerating smoothly and keeping the speed limit. Players can obtain more fuel upon crossing various "checkpoints" throughout the city. Non-eco-friendly driving behavior is penalized with more rapid decreases in the car's fuel supply (sometimes rendering them unable to get to the next checkpoint). Running a red light or bumping into other cars also reduces a player's fuel.

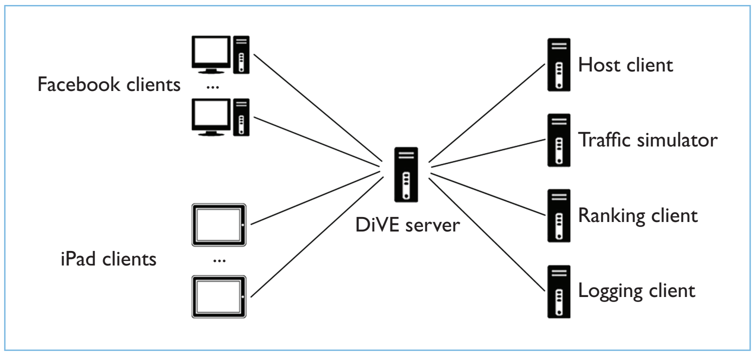

*Figure 2. $iCO_2$ architecture. The left side shows the clients controlled by the players (running $iCO_2$ on Facebook or iPad). The right side shows the clients required for the supporting tasks. Each client is connected to the DiVE server, which is responsible for all network communication in $iCO_2$.*

We created this game mode to target players' competitive nature. Players' scores are shared in an overall ranking, as in old-style arcade games.

In "Campaign" mode, users play the role of citizens with daily tasks. This mode is composed of simple missions, in which the challenge is to drive the car in an eco-friendly way from Point A to Point B. The level of challenge increases as players advance in their career mode. More complex missions require that players pick up passengers on the way, while simultaneously planning the shortest route to save fuel. Players must consider all requirements, while always bearing in mind eco-friendly driving techniques. In each mission, users also have side-challenges related to eco-friendly behavior, such as driving a safe distance from other cars and reducing their average fuel consumption. Depending on how players perform in these side-challenges — such as the percentage of safety distance maintained or gasoline consumed during a mission — they are awarded bronze, silver, or gold medals. These medals are converted to an overall medal for mission performance (see Figure 1d). The incentive to play in "Campaign" mode is to accomplish the missions with better results each time and obtain as many gold medals as possible, and to advance to more complex missions.

Score ranking and rewards-like medals are an "extrinsic" motivation to play the game; human beings are naturally competitive and like to compare their skills and performance with those of others.[4] We also added real-time challenge balancing (RCB) as an "intrinsic" motivation. RCB is a method to dynamically adapt the game task's difficulty, such as eco-driving, to the user's skill level, so that eco-driving is neither too difficult (causing frustration) nor too easy (causing boredom).[8] Simply speaking, the RCB method provides instructions to computer-controlled agents, such as traffic lights and NPC vehicles, to create traffic situations that make eco-driving more difficult. Those agents thus take the role of "opponents" that try to achieve the optimal challenge level for each user's skill (unbeknownst to the player). In this way, the player reaches a state of "flow" — that is, the player is immersed in a feeling of enjoyment and focus.[9] When not instructed by RCB, the NPC vehicles are controlled by a state-of-the-art traffic simulator.

## DiVE-Based System Architecture

Distributed Virtual Environments (DiVE) is our original middleware for real-time communication among clients. The communication is based on Photon (see www.exitgames.com), which makes it highly scalable, fast, and cross platform. As Figure 2 shows, with a DiVE server, multiple clients can easily share a virtual environment. Each client states which types of "entities" it's interested in (for instance, "cars"), and which entities it controls in the world (for instance, the player car).

We implemented the following functionality as DiVE clients:

- *$iCO_2$ client* — each instance of the $iCO_2$ application. Users can join the virtual city and play the game by logging in via Facebook Web client or iPad.
- *Host client* — the unique central manager for spawn points in the city. This client lets us avoid situations in which a player spawns (is created) in a position that another player occupies.
- *Traffic simulator client* — manages all the NPC cars and traffic lights in the virtual city.
- *Ranking client* — maintains a leaderboard for the "Free Drive" mode and holds the progress data for the "Campaign" mode.
- *Logging client* — constantly receives information about each player and logs it in a persistent database for later analysis.

A special feature of the DiVE-centric architecture is its extensibility. Each time we need new features that require network communication, we can easily develop a new DiVE client that implements the feature. Because it's simple to integrate a new client with DiVE, we

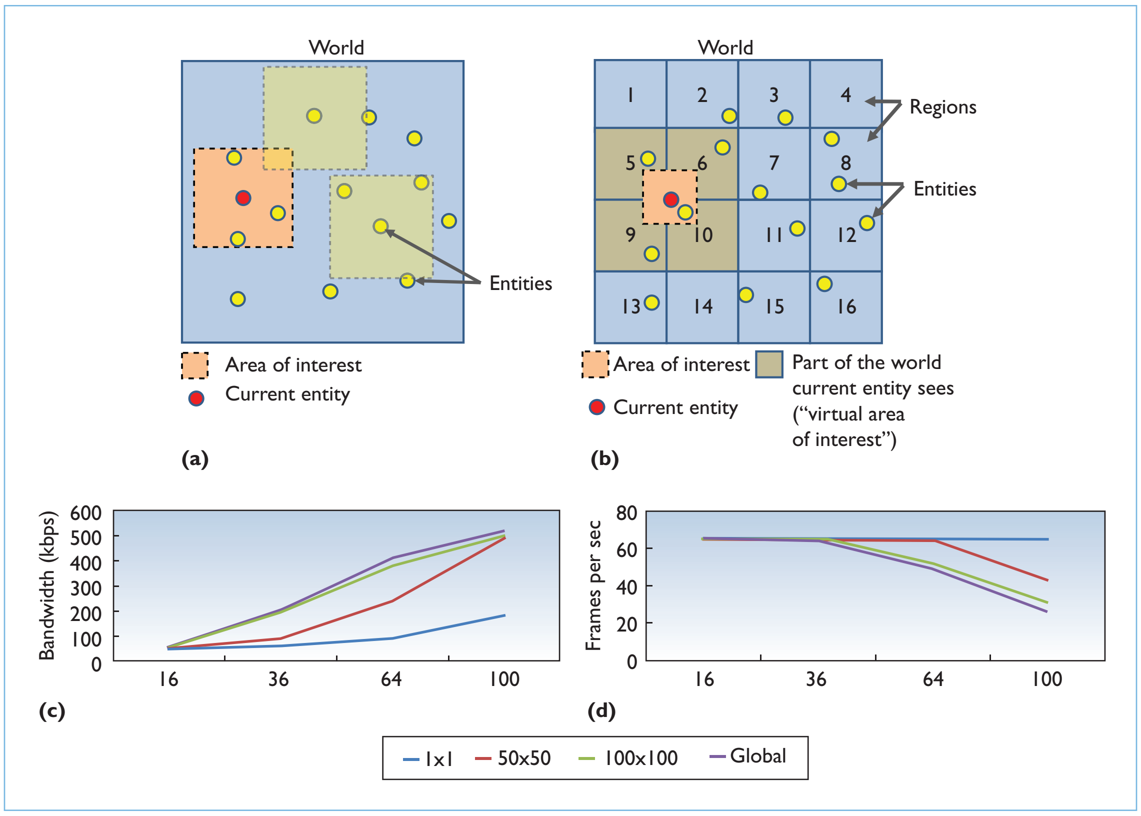

*Figure 3. DiVE's two approaches to continuous data communication. (a) The basic method focuses solely on the entity's area of interest. (b) The region-based method adds to the basic approach by dividing the world into regions. (c) The region-based method's results for bandwidth. (d) The region-based method's results for frames per second.*

can focus all our efforts on developing new features, rather than spending time implementing network communication.

DiVE also offers players a seamless interaction between devices. While driving on the iPad, the player can interact with all users who are playing on their iPads or using the Web player on Facebook. Because players are identified by their Facebook ID (even in the iPad version), they can switch versions without losing their game progress. For instance, users can complete a mission on the iPad. and later, when they play the game on Facebook, the mission will be marked as completed. Also, all stats and medals awarded are consistent across the platforms.

To support this functionality, DiVE relies on two technologies: Photon and Protobuf (code.google.com/p/protobuf). Photon provides the serialization for the object-based continuous data-sharing mechanism, whereas Protobuf is used for serializing objects in our events. These two technologies make DiVE compatible with virtually every platform supported by the .NET and Mono common language. Hence, we can easily port games that use DiVE to other platforms and still maintain seamless interaction between players on different platforms.

## Controlled Performance Tests of DiVE

In the naïve approach to continuous data communication, each entity is notified about every other entity in the networked virtual world. Each time an entity is updated, it notifies the server and the server notifies all other $n - 1$ entities. Thus, the number of communication messages are on the order of $O(n(n - 1))$.[10]

DiVE handles continuous data communication using an "area of interest" method.[11] An entity's area of interest, or *aura*,[12] is the portion of space the entity is aware of at a given time. DiVE offers two approaches: a basic method and a region-based method. With the basic method, the server must check which other entities are within the interest area and update them (see Figure 3a). Each entity's area of interest is a square tile.

In DiVE, we use the region-based method instead. This method, which is based on Photon, divides the world into regions (see Figure 3b). As in the basic method, each entity has its own area of interest. This approach's novelty is that this area isn't used to directly detect which other entities the entity should be aware of. Instead, it defines the regions to which the entity belongs. Whenever an entity enters a region, it receives updates of the region's other entities. An empirical comparison of both methods in a controlled setting using Matlab indicates a quadratic complexity of the basic method versus a linear complexity of the region-based approach.

We also evaluated DiVE's performance on key metrics such as bandwidth and frames per second (FPS) under differently sized interest areas. Other researchers have experimented with different shapes, such as triangle, hexagonal, and so on,[13] rather than different region sizes. In our tests, we used a $200 \times 200$ meter world, with a fixed region size of $50 \times 50$ m. Each client had exactly one entity that moved every 10 milliseconds for 30 seconds. The movement aimed to trigger the communication with DiVE. Whenever an entity is updated, DiVE is notified and propagates the event to every entity whose area of interest intersects with that of the moving entity.

First, we measured the bandwidth in four runs with differently sized interest areas and an increasing number of clients (see Figure 3c). With smaller interest areas, we obtain weakly ascending lines. Although the entities notify DiVE about their movements, DiVE doesn't need to notify many entities, because they don't intersect each other's interest areas. In our fourth run (thick black line), we used an area of interest that would guarantee that each entity would see every other entity. It's thus called "global" and corresponds to the naïve approach previously mentioned. It shows the worst performance.

Second, we tested the area of interest's impact on the FPS as an indicator of the smoothness of users' experience. We used a simplified version of a client that didn't perform logical or graphical computation. Instead, we simulated the work that a client would have to perform to handle the information about the entities (local and remote) by adding some arithmetic computations that increase with the number of entities. As expected, performance is worst when all areas of interest intersect each other (creating the global model; see the violet line in Figure 3d). On the other hand, when using smaller areas of interest ($1 \times 1$, $50 \times 50$, and $100 \times 100$), we can sustain higher FPS with an increasing number of clients. In particular, where the global model drops below 30 FPS, the area of interest models are still above this frame rate.

Even with a good performance, it's difficult to receive updates from other entities at a speed above 30 FPS due to network constraints and fluctuations. Hence, if we directly apply the movement updates in remote entities to each client's local representation of the remote entity, the player will notice degraded smoothness. Therefore, we perform a simple interpolation between each update — that is, we calculate the position an entity should occupy by interpolating the last two positions received from the network. We know that the entity occupies a position A in time $t$ and a position B in time $t'$. We can thus determine all the positions that the entity should occupy between $t$ and $t'$ by interpolating positions A and B. This method supports smooth movement, even in the presence of network constraints and fluctuations.

## Data Collection on iPads "In the Wild"

From a research perspective, iCO2's aims to provide valuable driving behavior data to traffic engineers. The logging client records each driver's time stamps, user ID, velocity, carbon emissions, remaining fuel, and ($x$, $y$, $z$) position. Logging user behavior helps us better understand the impact design decisions have on player behaviors. Specifically, we want to assess the effectiveness of the colored "frame" around the acceleration slider, which turns green to indicate eco-friendliness (as in Figure 1b) and red to indicate non-eco-friendliness. To calculate eco-friendliness, we use the Emissions from Traffic (EMIT) model.[14]

We collected driving behavior data from 97 distinct players using the iPad version of $iCO_2$. The players were recruited through a marketing initiative. To receive an award, players had to drive at least two minutes. The observed average driving duration was 3.57 minutes. In the study, we focused on users' eco-driving behavior in terms of smooth acceleration, which we defined as steady (constant) acceleration — that is, driving in which the slope of smooth acceleration is close to zero. If players drive in an eco-unfriendly way — that is, they accelerate or decelerate harshly or cross the speed limit of 50 km/h — they are prompted by a red frame on the acceleration bar.

To illustrate smooth acceleration, Figure 4 shows four line charts that plot 22 categories of acceleration slope, corresponding to the rate of change in acceleration in the positive direction (acceleration) or negative direction (deceleration). Based on maximum deceleration and acceleration (–6 m/sec$^2$ and 6 m/sec$^2$), we defined 22 category ranges: the first category range is [–infinity, –100), the second category range is [–110, –100), the third category range is [–100, –90), and so on. Each line chart is plotted on a cluster of players and represents a characteristic of smooth acceleration. The four clusters are generated by the *k*-means clustering algorithm[15] over smooth acceleration, where $k = 4$. *k*-means is an automated clustering algorithm that uses greedy search and a distance metric to partition data into *k* clusters. A data point is close to other data points in the same cluster and far from data points in the other clusters. The number of the cluster size ($k = 4$) is selected by reducing the sum square error (SSE) over *k*. We ran the *k*-means algorithm 19 times, with $k = 2$ to 20, and recorded their SSEs. We selected the best *k* on the first trivial reduction on the SSE value.

Figure 4 shows four line charts. Each point on a line chart shows the percentage of a slope of acceleration. We can roughly divide categories (ranges) into three types:

- the middle category (11th and 12th categories) represents smooth acceleration;
- the left-most category (1st category) represents very sharp deceleration (braking) or collision with another car or building; and
- the right-most category (22nd category) represents a sharp acceleration.

To ensure eco-driving, a player should stay in the middle categories.

As Figure 4 shows, the best eco-driving cluster was Cluster 2, whereas Cluster 4 had the worst performance. Cluster 2 demonstrated the most eco-friendly behavior because it had the highest percentage in the middle categories and a low percentage in the other categories. That is, Cluster 2 players mostly accelerated close to constant (smoothly). In contrast, Cluster 4 had the lowest percentage in the middle categories and the highest in the 1st and 22nd categories. In the middle categories (11th and 12th), Cluster 2's share was 64.5 percent, whereas Cluster 4's share was 30.2 percent. So, Cluster 2 players drove 2.13 times smoother than Cluster 4 players. We can confirm this result looking at Cluster 2's lowest average velocity (13.9 km/h) and average carbon emission (32.51 g/sec). However, there was also a difference in the number of players: Cluster 2 had 19 players, whereas Cluster 4 had 37 players.

Although the game's objective is eco-driving, 10 players seemed to drive as if it were a racing game. These players belonged to Cluster 3, as the variations in acceleration show. Cluster 3 had the highest percentage of adjacent categories (9th, 10th, 13th, 14th, and 15th categories) around the middle categories. Therefore, Cluster 3 had both the highest average velocity (62.25 km/h) and carbon emission (67.38 g/sec).

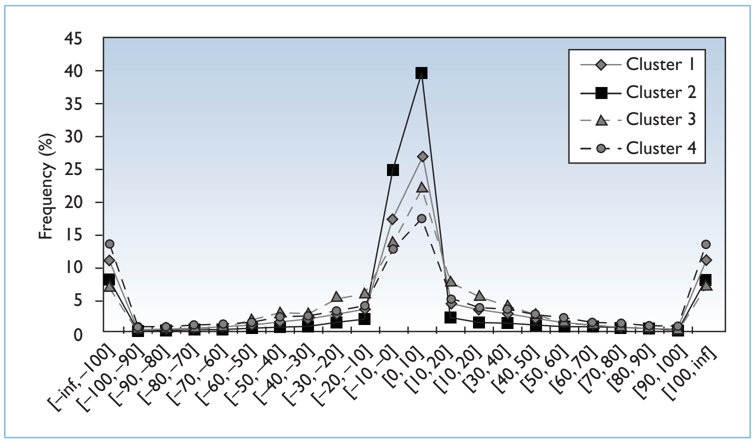

*Figure 4. Line charts showing characteristic driving behavior. Each point on a chart shows the percentage of a slope of acceleration.*

We've identified two key challenges for future work. First, we tested iCO2 "in the wild" with real users operating iPads. The data mining results showed that only one-fifth of the players followed eco-driving practices, which was the game's purpose. Although the result is specific to our game, it points to a general issue: How do you deal with players who don't comply with the game's purpose? Hence, one of our future opportunities is to use the knowledge obtained from offline data mining (such as clustering) to create online alerts. For example, if a player is identified as belonging to a "racer" cluster, we can send a warning message to that player.

Second, we tested our DiVE networking component in a controlled setting. DiVE's novelty is that it reduces network communication by integrating fixed regions and variable areas of interest. The results indicate that DiVE scales well to 100 simultaneous players when using a single DiVE server. Our challenge now is to accommodate additional simultaneous players. Therefore, we are working on a server communication solution.[16]

DiVE's future development will facilitate even larger "in the wild" studies with hundreds of simultaneous eco-drivers and thus let us investigate the effects of ubiquitous eco-driving. To attract such large numbers of players, however, we must also assess the effectiveness of our game mechanics. This could lead to further improvements of i$CO_2$'s playability.

i$CO_2$ is available on Facebook and iPad, and as a physical installation on the "Cube by QUT" at Queensland University of Technology's Science and Engineering Centre; for more information, see https://sites.google.com/site/ico2globallab.

**Acknowledgments**

We are grateful to Kamthorn Puntumapon for his help with the data analysis. This work is partly supported by a Grand Challenge grant and other grants from Japan's National Institute of Informatics; by a Kaken grant from the Japan Society for the Promotion of Science; and by Portugal's Fundação para a Ciência e a Tecnologia, under project PEst-OE/EEI/LA0021/2013.

## References

1. C. Aldrich, *The Complete Guide to Simulations and Serious Games*, Pfeiffer, 2009.
2. C.W. Thompson, "Next-Generation Virtual Worlds," *IEEE Internet Computing*, vol. 15, no. 1, 2011, pp. 60–65.
3. *PC Gaming. Power to the People*, trend report, New Zoo, 2014; www.newzoo.com/trend-reports/pc-gaming-trend-report-power-to-the-people.
4. W. Peng and J. Crouse, "Playing in Parallel: The Effects of Multiplayer Modes in Active Video Game on Motivation and Physical Exertion," *Cyberpsychology, Behavior, and Social Networking*, vol. 16, no. 6, 2013, pp. 423–427.
5. K. Gajananan et al., "An Experimental Space for Conducting Controlled Driving Behavior Studies Based on a Multiuser Networked 3D Virtual Environment and the Scenario Markup Language," *IEEE Trans. Human-Machine Systems*, vol. 43, no. 4, 2013, pp. 345–358.
6. H. Prendinger et al., "Tokyo Virtual Living Lab: Designing Smart Cities Based on the 3D Internet," *IEEE Internet Computing*, vol. 17, no. 6, 2013, pp. 30–38.
7. F. Bella, "Driving Simulator for Speed Research on Two-Lane Rural Roads," *Accident Analysis & Prevention*, vol. 40, no. 3, 2008, pp. 1078–1087.
8. M. Madruga and H. Prendinger, "iCO2: Multi-User Eco-Driving Training Environment Based on Distributed Constraint Optimization," *Proc. 12th Int'l Conf. Autonomous Agents and Multiagent Systems* (AAMAS 13), 2013, pp. 925–932.
9. M. Csikszentmihalyi, *Flow: The Psychology of Optimal Experience*, Harper Perennial, 1990.
10. H. Liu, M. Bowman, and F. Chang, "Survey of State Melding in Virtual Worlds," *ACM Computing Surveys*, vol. 44, no. 4, 2012, pp. 1–25.
11. J.-S. Boulanger, J. Kienzle, and C. Verbrugge, "Comparing Interest Management Algorithms for Massively Multiplayer Games," *Proc. 5th ACM SIGCOMM Workshop Network and System Support for Games* (NetGames 06), 2006; doi:10.1145/1230040.1230069.
12. S. Benford and L. Fahlén, "A Spatial Model of Interaction in Large Virtual Environments," *Proc. 3rd European Conf. Computer-Supported Work*, 1993, pp. 109–124.
13. K. Prasetya and Z.D. Wu, "Performance Analysis of Game World Partitioning Methods for Multiplayer Mobile Gaming," *Proc. 7th ACM SIGCOMM Workshop Network and System Support for Games* (NetGames 08), 2006, pp. 72–77.
14. A. Cappiello et al., "A Statistical Model of Vehicle Emissions and Fuel Consumption," *Proc. 5th Int'l Conf. Intelligent Transportation Systems*, 2002, pp. 801–809.
15. J. MacQueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Proc. 5th Berkeley Symp. Mathematical Statistics and Probability*, 1967, pp. 281–297.
16. P. Quax et al., "Dynamic Server Allocation in a Real-Life Deployable Communications Architecture for Networked Games," *Proc. 7th ACM SIGCOMM Workshop Network and System Support for Games* (NetGames 08), 2006, pp. 66–71.

**Helmut Prendinger** is a full professor at the National Institute of Informatics, Tokyo. His research interests include artificial intelligence, massively multiuser networked 3D virtual environments, intelligent transport systems (ITS), and affective computing. Prendinger has a PhD in logic and artificial intelligence from the University of Salzburg. He's a member of IEEE and ACM. Contact him at helmut@nii.ac.jp.

**João Oliveira** is a graduate student in software engineering at Instituto Superior Técnico–Lisbon. His research interests include 3D virtual environments, distributed computing, and multimodal interaction. Oliveira has a BSc in information systems and computer engineering from Instituto Superior Técnico, University of Lisbon. Contact him at joao.delgado.oliveira@tecnico.ulisboa.pt.

**João Catarino** is a graduate student at Instituto Superior Técnico–Lisbon and a research student at Instituto de Engenharia de Sistemas e Computadores–Investigação e Desenvolvimento (INESC-ID) in the Intelligent Agents and Synthetic Characters Group. His research interests include virtual worlds, artificial intelligence, game design, and development. Catarino has a BSc in information systems and computer engineering from Instituto Superior Técnico–Lisbon. Contact him at jcbpc@ist.utl.pt.

**Marconi Madruga** is a graduate student at the National Institute of Informatics, Tokyo. His research interests include artificial intelligence in games, human-computer interaction, multi-agent systems, and virtual characters. Madruga has an MS in informatics from the Graduate University for Advanced Studies (SOKENDAI), Japan. Contact him at koni.kun@gmail.com

**Rui Prada** is an assistant professor at the Computer Science Department of Instituto Superior Técnico, University of Lisbon, where he teaches courses on user-centered design, game design, and development and socially intelligent agents. He is also a senior researcher at INESC-ID in the Intelligent Agents and Synthetic Characters Group, where he develops social intelligent agents, virtual collaborative environments, user-centered design, and computer games. Prada has a PhD in computer science in the field of artificial intelligence from Instituto Superior Técnico. Contact him at rui.prada@tecnico.ulisboa.pt.